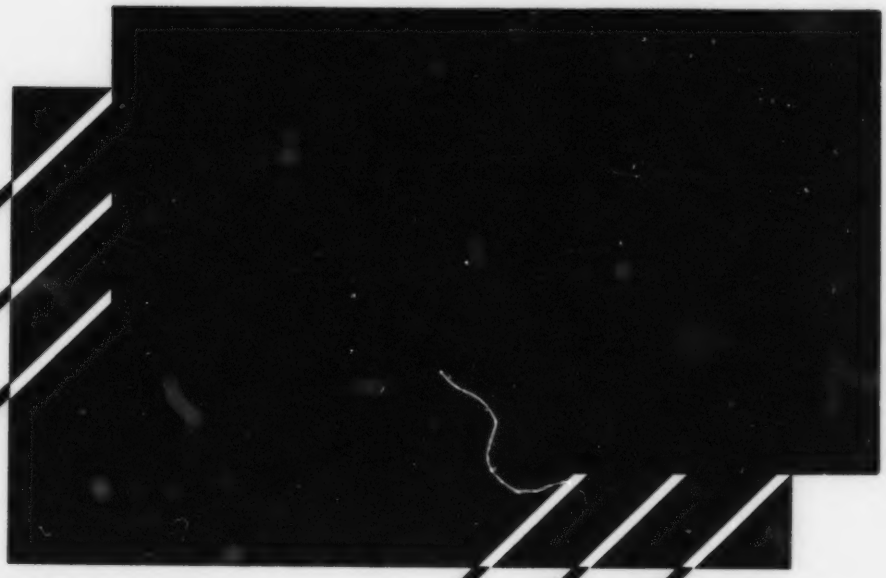
Gouvernement du Canada
Ministère des Communications

Government of Canada
Department of Communications

Le Centre canadien de recherche sur l'informatisation du travail
Canadian Workplace Automation Research Centre



Canada

92-05730/

Finite-state Morphology: Overview of Existing Models and Applications in Continuous-text Environments.

Michel Simard

Assisted Translation Group

Canadian Workplace Automation Research Centre

Communications Canada

Laval

may 1990

cat. N° Co 28-1/51-1990 E
ISBN 0-662-17911-0

The author is solely responsible for the opinions expressed in this report.

* Ce rapport est aussi disponible en français.

Finite-state Morphology: Overview of Existing Models and Applications in Continuous-text Environments.

Michel Simard[1]

Centre canadien de recherche sur
l'informatisation du travail
Communications Canada
1575, boul. Chomedey
Laval (Québec)
CANADA H7V 2X2

Abstract

Finite-state morphological models are formalisms for describing the set of valid word-forms of a natural language. Being well-suited for computer implementations, they have typically been used for creating systems that efficiently recognize and generate isolated word-forms. In this paper, we give a presentation of Koskenniemi's *two-level* morphological model, followed by a comparison with alternative approaches. Integrating such models to natural-language processing systems that deal with NL *sentences* typically implies modifying the morphological component so that it works in a *continuous-text* environment. We discuss how this transition from isolated-words to continuous-text may be done, and show that, in addition to the orthographical phenomena normally described by morphological models, the resulting system displays interesting properties for describing "inter-word" phenomena such as *elisions* and *contractions*.

Topics: two-level morphology; finite-state morphology; computational linguistics.

Introduction

The interest in computational models of the morphology of natural languages is quite recent. It may be traced to the appearance of Kimmo Koskenniemi's 'two-level' model ([Kos83]), a linguistic model for morphological analysis and synthesis that is suitable for a computer implementation. It consists in a formalism for describing how words of a language are constructed from a set of morphemes. This description may then be used to generate a program that analyzes or synthesizes words of that language. It has been implemented in Pascal ([Kos83]), in Lisp ([Kar83],[DKK87]) and in Prolog ([Boi88b]), among other languages. A number of computational systems have since emerged which, given the 'surface form' (physical appearance) of a word, return a description of that word

1. Email : m_simard@ccrit.doc.cdn

derived from its decomposition into morphemes, and vice-versa. These systems are usually referred to as "finite-state models" because they use finite-state devices as parsers. Such formalisms are well suited for designing computer applications requiring lexical descriptions of highly inflected languages, such as Finnish or Turkish, as well as for 'simpler' languages such as English or French

Many of these finite-state models were originally designed to handle isolated words. When examining the possibility of integrating such models to larger natural-language processing environments (e.g. an automatic translation system, or a natural-language database interface), the following question naturally arises: what happens when finite-state morphological models are modified to deal with *continuous-text*, as is the case if they are to be used within a system that works with natural language *sentences*? Interestingly, it seems that the resulting systems are capable of handling things that most syntactic models are not very good at, and that morphological models were not primarily designed for. There are a number of phenomena that seem to be conditioned by the relative position of a word within a sentence, rather than by its syntactic role. Examples of this are elisions and contractions, liaisons in speech, or simply the appearance of 'spaces' in written text. While these phenomena are usually difficult to describe syntactically, they fall within the field of competence of one of the components of the two-level model — the one that deals with *phonology* or *orthography*.

In the first part of this paper, we examine Koskenniemi's two-level model in detail: in section 1, we present some basic concepts, then discuss the formalism in section 2, and finally consider its implementation in section 3. In section 4, we review a number of other approaches that have emerged since Koskenniemi's. The second part of the paper deals with the continuous-text question: in section 5, we give a brief sketch of how Koskenniemi's two-level rules may be used to describe *inter-word phenomena*.

1 Some Background

In general, a lexicon may be seen as defining the relation between the *surface forms* of the words of a language (their physical appearance in written text or speech) and their *lexical descriptions* (or *definitions*). In a computational perspective, if both sets of possible surface forms and of possible lexical descriptions are finite, then one way of implementing this relation is to create an exhaustive list of *surface-form/lexical-description* pairs that satisfy the relation, and to use some searching algorithm. If one or both sets turn out to be infinite, or simply of relatively large size, this solution becomes unacceptable for obvious reasons.

The key idea behind Koskenniemi's system is that this 'lexical relation' may be decomposed and described entirely through the descriptions of the *morphology* and *phonology* of the language, and that from these, efficient recognition/generation programs may be obtained. The point here is that in the general case, the description of these individual relations is considerably more compact than the exhaustive list suggested above, and, needless to say, finite.

Informally, morphemes are the objects words are constructed from: it is generally agreed upon that all words of a language are made up of smaller pieces, put together ac-

cording to rules that define the *morphology* of the language. For example, a word such as *transported* can be seen as formed of three morphemes: prefix *trans-*, root *port*, and suffix *-ed*. A distinction is usually drawn between *derivational* and *inflectional* morphologies: 'derivation' refers to the morphological processes by which new words are obtained from older ones (e.g. *transport* from *port*), and 'inflections' refer to the various shapes that a same and unique word may take (e.g. *transport*, *transported*, *transporting*, etc.)[2].

This is further complicated by the fact that the same morpheme may 'realize' in several ways (appear under various shapes on the surface) depending on its environment, i.e. on the shape of the morphemes next to which it appears. These variations are controlled by the *phonology* of the language. In general, 'phonology' refers to these phenomena, as observed in spoken language. While it is generally accepted that phonological phenomena in speech obey fairly strict principles, their counterpart in written text (sometimes called 'orthography' in the computational linguistics literature) is seldom studied by linguists because in most languages, spelling conventions seem rather arbitrary. In spite of this, in order to simplify all that follows, we will focus our attention on orthography rather than phonology. As well, we will try as much as possible to stick to English in the examples.

2 The Formalism

Koskenniemi's idea is to "split the work in two", i.e. to introduce an intermediate representation, called the *lexical form*, between the surface form and the lexical description. This representation is actually the result of a concatenation of objects representing morphemes of the language. When verifying if some surface-form/lexical-description pair is in the 'lexical relation', we check that there exists a lexical form that satisfies a first relation with the surface form (the orthographical relation) and a second relation with the lexical description (the morphological relation)

The first underlying hypothesis here is that the lexical description of a word may be easily computed as a function of the descriptions of its composing morphemes[3]. To obtain these, it must be possible to decompose a word into morphemes: the second hypothesis is that this decomposition is easy to do if we have access to a representation of the word displaying it as it would appear if the orthographical process did not occur, i.e. as if orthographical rules did not affect the surface realization of this word: this is what Koskenniemi's intermediate representation corresponds to. The third hypothesis is that these orthographical rules are easy to apply and *un*-apply, so that the correspondence between the lexical and surface form is also easy to compute. The morphological relation is defined by means of a morphological lexicon, i.e. a lexicon of all morphemes of the language, along with morphological rules (which morphemes may combine with one another, and in what way), and the

2. Actually, things are not all that simple, mainly because it is not clear what exactly is meant by 'two different words' and 'the same word under two different forms', but these problems need not concern us here.

3. When using the expression "easy to compute", we refer to the intuitive notion of 'practical efficiency' rather than to that of 'tractability': while the general problem of parsing the languages defined by Koskenniemi's model has proven to be NP-complete (see [Bar86]), in most real-life cases, efficient parsers may be constructed.

orthographical relation with a set of *two-level* orthographical rules. These two components are described in more details below.

2.1 The Lexicon

In the two-level model, the lexicon formalism is used to define the morphemes and the morphological rules of the language. It appears as a list of entries, each of which corresponds to a morpheme, and is partitioned into *sublexicons* on the basis of the a set of principles called the *morphotactics* of the language (this partitioning of morphemes is akin to the *categorization* of words in the syntax). To each morpheme is associated a *lexical form* (its appearance in the intermediate lexical representation), a *continuation class*, and some attached information.

The lexical form of a morpheme is somehow meant to denote its *underlying canonical form*, i.e. the unique representation from which all its surface realizations are derived, according to the orthographical rules of the language. It is up to whoever is producing a lexicon to decide what these lexical forms should actually look like, but in most cases, we can expect them to differ only in minor ways from their surface counterparts: a lexical form should be something like a *generalization* of all the surface realizations of the morpheme to which it is attached. Therefore, the *lexical alphabet* of a language, i.e. the alphabet in which these forms are written, although it may be anything, will probably contain all characters of the *surface alphabet*, plus maybe a small set of additional characters (diacritics, etc.). For example, the lexical form of the English verb *to carry* could be something like *carrÿ*, where character *ÿ* is meant to denote a generalization of surface realizations *y* (as in *carry*) and *i* (as in *carried*). Yet another possibility is simply to use one of the surface realizations: in this case, *carry* would also be an acceptable lexical form for the verb *to carry*, insofar as both *y* and *i* were considered as possible surface realizations of lexical character *y*.

Each *continuation class* refers to the set of sublexicons that contain morphemes which may concatenate to the right of the current one: they are the means by which the *morphological rules* of a language are defined in this model. Some sublexicons are labeled as *root*: those contain morphemes that may 'begin' words of the language. There is also a special *final* continuation class, which refers to an empty set of sublexicons, and which is given to morphemes that may 'end' a word. Lexical forms of words of the language may then be obtained by concatenating morphemes of the lexicon in ways prescribed by the continuation classes, starting with a 'root' morpheme, and ending with a 'final' continuation class morpheme.

Such a lexicon for a (very) small subset of English appears in figure 1. Boxes denote sublexicons, ellipses denote continuation classes, morphemes are in italics (actually, the lexical forms of these morphemes), and *Vroot* and *Nroot* are the root sublexicons. Morphemes that do not point to a specific continuation class are considered to be 'final'. From this lexicon could be obtained such 'words' as

mice *eat0* *cry+ed* *walk+er+s+'s*

Notice that in practice, the word 'morpheme' may be taken in a very loose sense when building a lexicon. For example, if it turns out to be convenient in a given context to

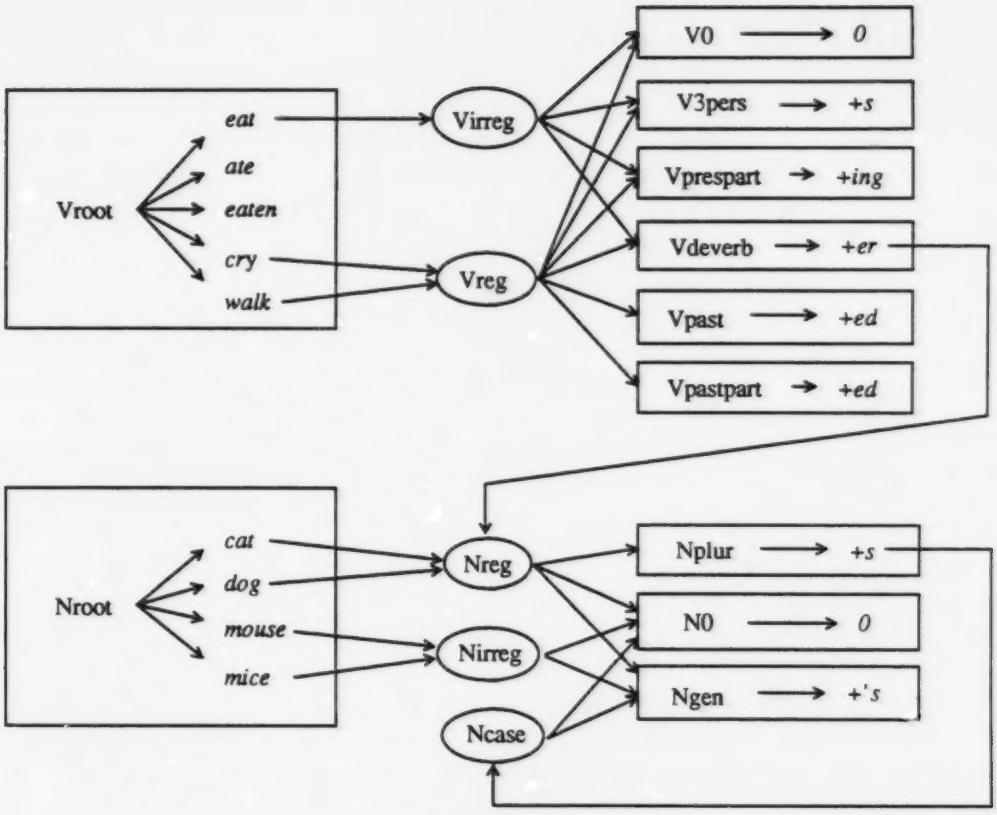


figure 1 Sample English lexicon.

consider the expression *transporter room* as a 'morpheme', then so be it. From a purely linguistic point of view, this may be a bit shocking, but as it happens, it is common practice.

Here, the information attached to the morphemes has been omitted for clarity, but in the original model, it appears as a string of text. The lexical description of a word is the result of the concatenation of the individual descriptions of its morphemes in their order of appearance. For example, if morpheme *cry* has attached information "*verb 'to cry'*" and suffix *+ed* has "*past-tense*", then lexical form *cry+ed* has lexical description "*verb 'to cry' past-tense*". This rather primitive way of 'collecting' lexical information is a theoretical expedient whose sole purpose is to simplify the demonstration of the correctness of Koskenniemi's analyses. In later implementations of the model, the format of lexical descriptions as well as the manner in which the description of a morpheme affects that of the words it appears in are left for the user to define.

2.2 Two-level Rules

Two-level rules define the relations that exist between the lexical and surface forms by specifying *character-to-character* correspondences between the two representations. In this way, they define the orthographical rules of the language. They have the following for-

mat:

<cp> <operator> <lc> — <rc>

where *<operator>* is one of ⇒, ⇐ or ⇔, *<cp>* is a *concrete pair set* (or *CPS*), and *<lc>* and *<rc>* are *regular pair expressions* (or *RPEs*), i.e. regular expressions over an alphabet of CPSs. These CPSs are pairs *(x,y)*, where *x* and *y* are elements of the lexical and surface alphabets respectively, and that describe possible realizations of lexical characters on the surface. Thus *(x,y)* may be read as "lexical character *x*, and its surface realization *y*".

Some characters have a special meaning within CPSs: the = stands for 'anything' as in *(=,s)* , to be read "any lexical character and its surface realization *s*". Conversely *(s,=)* stands for "lexical character *s* and whatever its surface realization is". Character ε stands for the 'null' character: *(s,ε)* reads "lexical character *s*, which realizes as the null string (nothing) on the surface". We also sometimes make use of variables: capital letters will normally be used for those, as in *(V,x)*, *V* ∈ { *a,e,i,o,u* }. Finally, when no confusion is possible, a single character *x* is used to denote a pair *(x,x)* where the lexical character is realized as itself on the surface (other notational shortcuts exist, but we will restrict ourselves to these for now).

The *<cp>* is the *correspondence part* of the two-level rule, i.e. the part that specifies the particular pair that the rule is concerned with, and *<lc>* and *<rc>* are the *left* and *right contexts* respectively, i.e. the parts that specify the context within which the rule applies. The *<operator>* specifies the *type* of the rule, as described below.

Context restriction rules, which take the form

<cp> ⇒ <lc> — <rc>

specify a context *<lc> — <rc>* within which correspondence *<cp>* is *allowed* to take place. In other words, if *<cp>* is *(x,y)* and *x* appears in a lexical form surrounded by *<lc>* on the left and *<rc>* on the right, then it *may* realize as *y* on the surface (we say that some pair *(x,y)* is *surrounded* by *<lc>* and *<rc>* if whatever appears immediately to the left and right of *(x,y)* is matched by regular expressions *<lc>* and *<rc>* respectively). In general, for a pair of lexical and surface forms to be orthographically correct, every one of its individual lex-

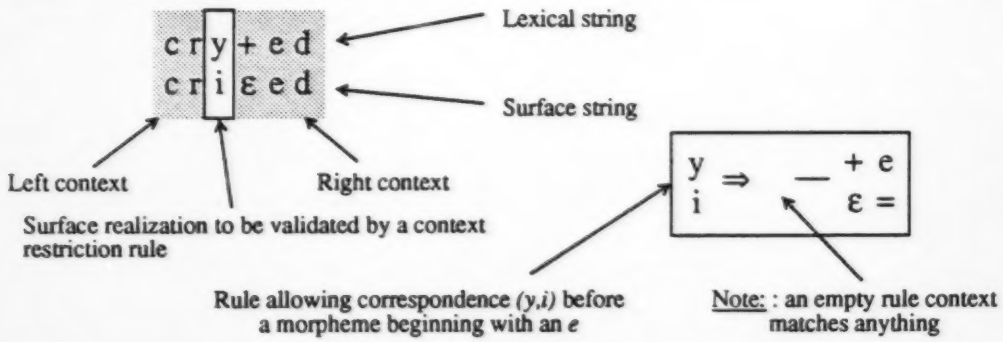


figure 2 Pairofcorrespondinglexicalandsurfaceforms,andmatching context restriction rule.

ical/surface character pairs must be validated by a context restriction rule. This is illustrated in figure 2. Notice how CPSs are written vertically, with the lexical character on top: this is the standard notation (but it doesn't fit that well in written text, so we alternate between the two notations).

Surface coercion rules take the form

<*cp*> ⇐ <*lc*> — <*rc*>

and specify the context within which a correspondence <*cp*> is *forced* to take place: if <*cp*> is *(x,y)*, and *x* appears within a lexical string surrounded by <*lc*> and <*rc*>, then it *must* realize as *y* on the surface. For a particular lexical/surface forms pair to be orthographically correct, none of its individual character pairs must contradict any of the surface coercion rules. What is understood by a 'contradiction' is illustrated in figure 3. Note that the existence of a surface coercion rule to force some correspondence *(x,y)* does not imply that this correspondence is *allowed*. What such rules say is something like "No matter what *x* is normally allowed to realize as, in this context, it must *not* realize as anything other than *y*". As a result, this type of rule turns out to be useful mostly to specify *disallowed* correspondences, as in the rule:

$$\begin{matrix} a \\ -b \end{matrix} \Leftarrow c — d$$

which reads "Surrounded by *c* and *d*, lexical character *a* must realize as a character that is not equal to *b* (*a may not* realize as *b*)". Therefore, such a rule forces a given correspondence *not* to take place. It is then a matter of checking matching context restriction rules to find what *a* may realize as in the context.

Composite rules are used to specify correspondences that are both allowed and forced. They are obtained by combining a context restriction rule and a surface coercion rule that share the same correspondence part and left and right contexts:

<*cp*> ⇔ <*lc*> — <*rc*>

is equivalent to

<*cp*> ⇒ <*lc*> — <*rc*> and

<*cp*> ⇐ <*lc*> — <*rc*>.

Full descriptions of the orthography of a language are, in principle, collections of

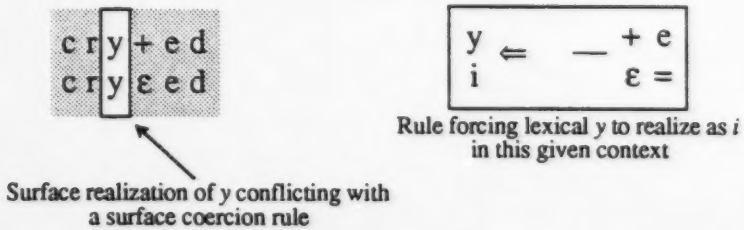


figure 3 Example of lexical/surface pair ruled out by a surface coercion rule

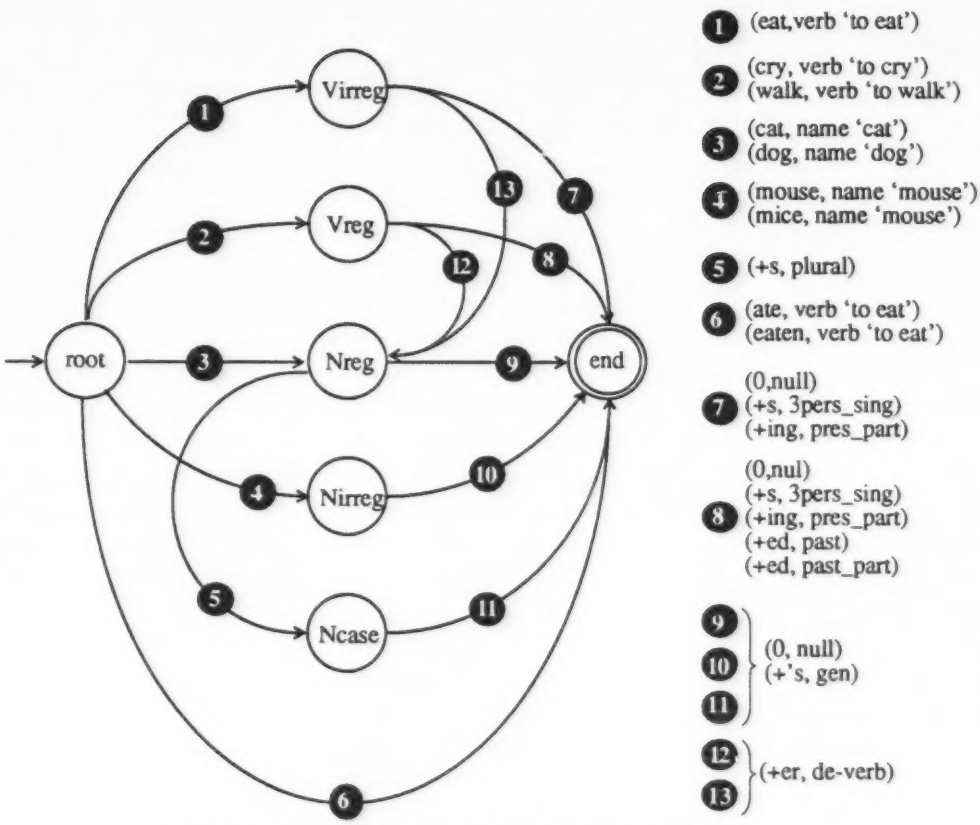


figure 4 Transducer corresponding to lexicon of figure 1.

two-level rules. For a given character pair, all context restriction rules are taken *disjunctively* (*one* of them must be satisfied, i.e. match the current character and context), and this disjunction is then taken *conjunctively* with all surface coercion rules (*all* of them must be satisfied, i.e. not be contradicted if they match the current context and lexical character). For a pair of surface and lexical forms to be accepted by the set of rules, this logical statement must be satisfied at every position in the string. Since the rules simply state relations between the two levels of representation, the formalism is not biased towards analysis or synthesis, and may be used for both.

3 The Implementation

The key idea behind the implementation of Koskenniemi's model is that both its formalisms (lexicon and two-level rules) define regular languages, for which efficient parsers are relatively easy to construct: individual finite-state devices corresponding to the lexicon and two-level rules may be produced and then used in parallel to analyze or generate surface forms. How this is done is discussed below.

A *finite-state transducer* is a special type of finite-state automaton that recognizes a

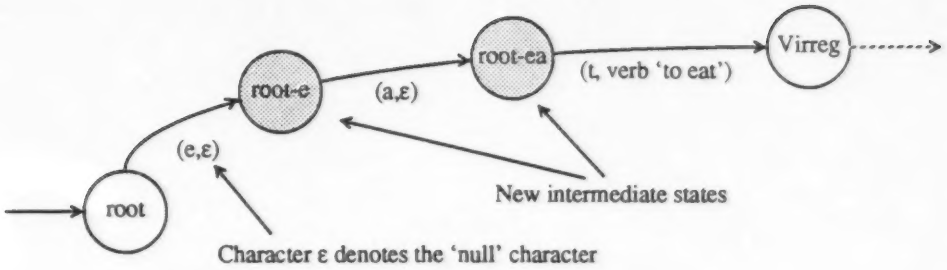


figure 5 Expansion of a transition of the lexical transducer.

language over an alphabet of pairs of characters. In formal terms, if A and B are alphabets, then an automaton that recognizes some language over $(A \times B)^*$ is a transducer. This type of computational device is interesting for us because it allows to describe relations between strings over different alphabets, insofar as the computation of these relations only requires a finite amount of memory. As it turns out, this appears to be the case for the relations that exist between lexical descriptions and lexical forms, and between lexical and surface forms.

If we take a particular lexicon under Koskenniemi's formalism, and consider the set of all morphemes as an alphabet (i.e. every morpheme of the lexicon is a 'symbol' in the alphabet), then it is obvious that the morphological rules, as encoded with continuation classes, define a regular language over this alphabet. Similarly, if lexical descriptions of words are obtained by concatenating descriptions of individual morphemes, then the morphological rules also define a regular language over the alphabet of descriptions attached to morphemes. From thereon, it is easy to see how a transducer that recognizes precisely all lexically well-formed morpheme/description pairs may be constructed. Such a device corresponding to the lexicon of figure 1 appears in figure 4 (drawing conventions of [HU79] for finite-state devices are used throughout this text). It is then a trivial matter to transform this transducer into one that operates on lexical forms instead of morphemes (see figure 5), so that we obtain an efficient computational device to recognize well-formed pairs of lexical descriptions and lexical forms.

Of course, we are interested in doing more than simply recognize well-formed pairs: we also want to perform *analysis* and *synthesis* operations. In formal terms, if we have a transducer M recognizing some language $L(M)$ over $(A \times B)^*$, then an *analysis* of some string s of A^* is some other string t of B^* such that there is a string x of $L(M)$ with projection on A $\Pi_A(x) = s$, and projection on B $\Pi_B(x) = t$ (*synthesis* is defined in a symmetrical fashion). Efficient graph-searching methods exist to compute such functions with transducers, and we will not discuss these here.

Two-level orthographical rules define a language over the alphabet of pairs of lexical and surface characters, which may also be recognized by transducers, although their construction is not as obvious as that of the lexicon's. As a matter of fact, the method by which these devices are produced from sets of two-level rules is a complex one for which, to our knowledge, no clear and formal description exists. Essentially, a set of two-level rules may be partitioned into subsets of rules sharing the same correspondence part (<*cp*> part), and

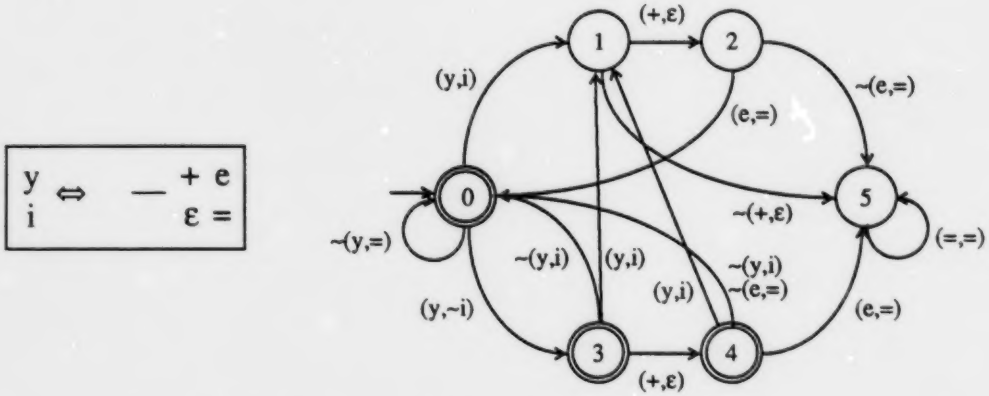


figure 6 Two-level composite rule, and corresponding transducer.

for each of these subsets, a transducer may be constructed such that a string over lexical and surface pairs of characters is accepted as orthographically correct if and only if it is recognized by all transducers.

Such a transducer corresponding to the composite rule responsible for the *(y,i)* correspondence in English appears in figure 6 (accepting states are doubly circled). Here, symbol = within a pair of characters stands for "anything" (as before), and symbol ~ stands for a 'negation'. For example, the pair *(y,~i)* refers to lexical character *y* realized as anything ***but*** *i* on the surface, *~(y,i)* refers to all pairs not equal to *(y,i)*, and *~(y,=)* will match any character pair whose lexical character is not *y*, no matter what its surface character is. The transducer is constructed in such a way that starting from state *0*, any string will be accepted, except those containing an occurrence of *(y,i)* that is not immediately followed by *(+,ε)* and *(e,=)*, and those containing a lexical *y* not realized as *i* that is immediately followed by *(+,ε)* and *(e,=)*. This reflects the intention of the rule, which says that *y* ***must*** realize as *i* in the given context, and that as it is the only rule concerning correspondence *(y,i)*, it is also the only context within which it ***may*** appear. If there was another rule concerning this correspondence, then ***both*** rules would have to be implemented within the same unique transducer.

Actually, since regular sets are closed under intersection, it is possible to combine all transducers obtained from a set of two-level rules into a single one. In theory, this produces a finite-state device whose number of states is a multiplicative factor of the number of states of the individual components from which it was obtained. However, the nature of the 'orthographical problem' is such that the resulting transducer may usually be optimized into one with a number of states closer to the sum than to the product, which makes the operation interesting if efficiency is a concern.

So ultimately, if we combine transducers as described above, we end up with just two transducers: one defining a language over pairs of lexical and surface characters (the 'orthographical language'), and one over pairs of lexical descriptions and lexical forms (the 'morphological language'). How these devices interact with one another varies with imple-

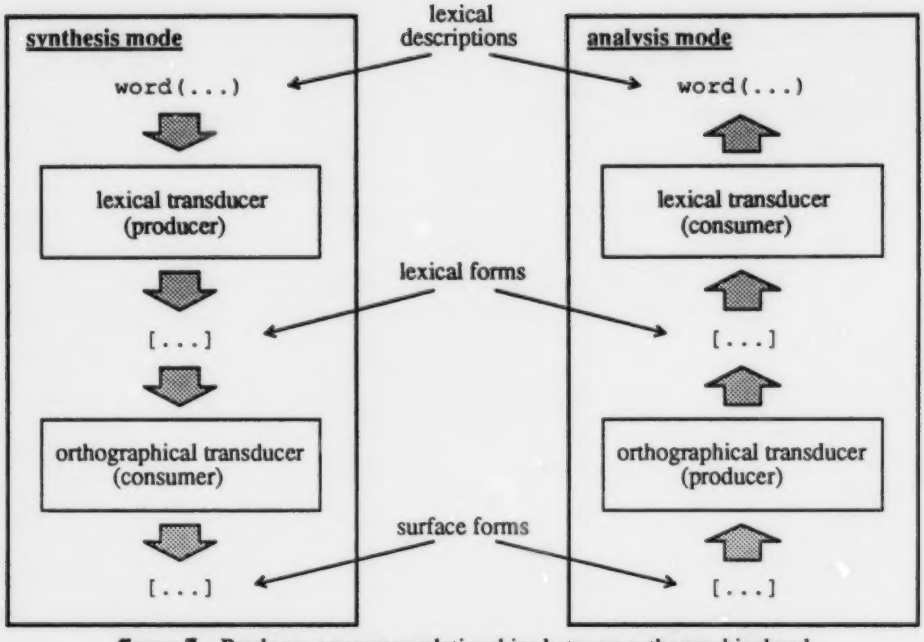


figure 7 Producer-consumer relationships between orthographical and lexical transducers.

mentations, but we can imagine some sort of 'consumer-producer' relationship to exist between the two (figure 7), so that one 'consumes' the lexical string as it is 'produced' by the other (which one is 'producer' and which one is 'consumer' is determined by whether the system is working in analysis or synthesis mode). The resulting system is a left-to-right parser which, depending of course on the exact nature of the lexicon and morphological and orthographical rules, can turn out to be quite efficient, both in analysis and synthesis.

4 Other Approaches

Being one of the few computational models for morphology and orthography to be of some interest from a linguist's point of view, and also the best known, Koskenniemi's two-level model has been the object of various criticisms. Let us have a look at what the major subjects of disagreement are, and at some of the alternative solutions proposed.

- *Encoding of Morphological Rules:* The two-level model is often criticized for the 'weakness' of its encoding of morphological rules: although not proven insufficient, it appears to be all too often inadequate to describe morphological phenomena other than suffixations and prefixations. Dolan ([Dol88]) lists such things as *reduplications* (repetition of a morpheme within a word), *infixations* (insertion of a morpheme 'within' another one) and *circumfixations* ('surrounding' of a morpheme by another one), while Anderson ([And88]) and Boisen ([Boi88a]) mention 'non-affixal' phenomena (e.g. 'ablaut' relations in some English verbs: *sing*, *sang*, *sung*) as other sources of problems. Typically, encoding these within the two-level formalism will require a lot of *lexicalization* (e.g. entering all

forms of the verb *to sing* as distinct morphemes in the lexicon) and reduplication of information (e.g. denoting the optional combination of elements of some sublexicon with a 'circumfix' requires the duplication of that whole sublexicon).

This has been improved in later presentations and implementations of the model, notably by providing more sophisticated methods of encoding and constructing lexical descriptions (feature structures, etc.; see [DKK87]), but many suggest that the problem comes mainly from the fact that continuation classes form a regular 'skeleton', and opt for a 'phrase structure' rule formalism, an idea first suggested by Karttunen & Wittenburg [KW83]. For example, Bear describes morphological rules using a PATR-type of formalism ([Bea86]), and Boisen's formalism is based on DCGs ([Boi88a]).

• *Parallel Application of Orthographical Rules*: Traditional models of generative phonology describe phonological phenomena using rewrite rules, meant to be applied 'in cascade', i.e. each rule applying on the output of the preceding one, and feeding its output to the next. Correspondences between the surface form and the underlying lexical form are then obtained by passing through several layers of intermediate representations, corresponding to the state of affairs between applications of rules. One hypothesis at the basis of Koskenniemi's model is that all intermediate levels between the surface and lexical forms may be bypassed, and correspondences between the two levels described directly with rules that apply simultaneously.

This view is not shared by all, and some models still favor the older method. Hankamer for example, suggests a parsing method for Turkish which, while being very close to that of Koskenniemi in its encoding of morphological rules, advocates a cascade application of orthographical rules (see [Han86]).

• *Separation between Orthography and Morphology:* One of the most important principles underlying Koskenniemi's model is the assumed separation between orthographical and morphological affairs: all that orthographical rules should know about higher levels (morphology, syntax, etc.) is communicated through the intermediate lexical representation, and vice-versa. This view is often challenged, as can be seen from the following examples.

In a model that is otherwise very close to Koskenniemi's, Bear ([Bea88]) suggests attaching what he calls 'negative rule features' to morphemes of the lexicon. What these features actually do is 'locally disable' certain orthographical rules so as to prevent them from being applied when 'exceptions' occur. This means that the morphological component has direct control on the behavior of the orthographical component. Note that this does not produce a formalism more powerful than Koskenniemi's: using negative rule features is absolutely equivalent to putting annotations in the lexical string. However, it does provide the advantage of not having to consider exceptions when writing orthographical rules, but only when writing the lexical entries to which these exceptions apply.

In a similar vein, Hankamer ([Han86]) uses different sets of rules for *roots* and for *suffixes*, thus assuming that orthographical rules 'know' what type of morpheme they are dealing with. This is actually just another way of locally disabling selected rules, only more

restricted than Bear's.

Cornell's IceParse model of inflectional parsing ([Cor88]) is a clearer departure from Koskenniemi's approach: in his view, morphology may be seen as some sort of 'orthographical phenomena' (as in e.g. 'umlaut' and 'ablaut' relations). Therefore, suffixes do not have corresponding lexical entries in his model, morphological rules are denoted the same way orthographical rules are, and define relations between the same two levels of representation, i.e. the surface form and some underlying representation akin to Koskenniemi's lexical form. The difference between the two types of rules is that morphological rules have the ability to affect the lexical descriptions of the lexemes to which they are applied. At the implementation level, both types of rules are translated into finite-state transducers.

At the other end of the spectrum lies Boisen's DCM formalism, where orthographical rules, encoded as transformational rules, are applied 'within' morphological rules. In other words, the application of a particular orthographical rule, in Boisen's view, is always a consequence of the application of a specific morphological rule.

5 Using Koskenniemi's Formalism in a Continuous-text Environment.

As mentioned earlier, when considering the integration of a formalism such as Koskenniemi's to a natural-language processing system, we are faced with the following situation: while the two-level model was originally designed to deal with single isolated words, we want it to work in a environment of continuous-text, i.e. one where borderlines between words are not always clearly marked. For example, while in most written languages the normal boundary character between words is the 'space', it is likely that things like *British Columbia* and *Department of Agriculture* will be viewed as words of their own, not to be decomposed in any way. This means that in addition to recognizing words, we have to deal with the problem of *isolating* them in the text.

In a single word context, a program implementing the two-level model is normally asked questions of the type "Give a lexical description of the following word if one exists" (the 'analysis question') or "If possible, produce a word having this lexical description" (the 'synthesis question'). It seems quite obvious to us that a program capable of answering these questions, provided with just a little extra information, may be used to answer questions about *sentences*, such as "Find all words of the language that match the left-hand side of this sentence, along with their corresponding lexical descriptions", or "Produce a word with this lexical description at that position of the sentence". Here, the word 'sentence' should be taken to mean simply 'list of words', and what the 'little extra information' required actually corresponds to is a description of how words appear within sentences (what the word boundaries look like, etc.).

Our claim is that this information about sentences may be encoded as *morphological* and *orthographical* information. This may be done by forcing the appearance of an explicit 'word-boundary character' at the end of each word of the language — a trivial matter under the two-level formalism's encoding of morphological rules if this character is seen as a morpheme — and by controlling its surface realization 'orthographically' with two-level

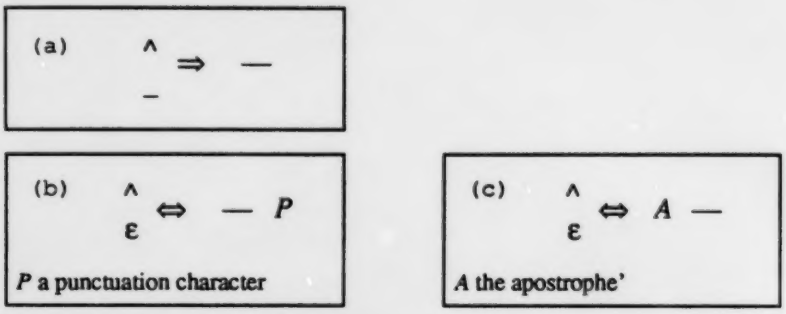


figure 8 Two-level rules sanctioning the realizations of word-boundary character '^'.

rules. Without going into too much detail, we can see how this may be done: let us take character '^' to be the *word-boundary* symbol. In general, it will realize on the surface as a space (that we will denote by '_'). However, there is a certain number of situations where such space must not appear, for example before a punctuation character, or after an apostrophe. This may be taken care of by a set of two-level rules, as shown in figure 8. Rule (a) says that '^' may always realize as a space '_' on the surface. However, other rules specify situations where it may and must realize as the null character ε: before a punctuation (rule b) and after an apostrophe (rule c).

One thing that should be noted is that this will work only if orthographical rules are applied to *whole sentences*, rather than to individual words: in general, the realization of a word boundary will be conditioned by the appearance of both what precedes it and what follows it. When integrating the two-level formalism to a natural-language processing system, one of our major concerns will be to allow for this 'global' application of rules.

Interestingly, it seems that besides standard 'intra-word' orthography, the surface appearance of word-boundaries is not the only thing that may be treated orthographically if two-level rules are applied in this way. For example such things as *elisions* and *contractions* could also be labeled as 'inter-word orthographic phenomena'. Let us have a quick look at this: in French, a good number of words take a different form whether they appear in front of a vowel (or mute *h*) or a consonant. For example, the *e* of word *le* (definite article or pronoun) disappears and is replaced by an apostrophe (') before a vowel, as in *l'avion* (a similar thing occurs in English with indefinite article *a*). This phenomenon is hard to describe syntactically, because it does not seem to depend in any way on the syntactic relation between the elided word (in this case, *le*) and the word which caused the elision.

The two-level rules of figure 9 solve the problem, not only for *le* but also for a bunch of others such as *ne*, *se*, *me*, *te*, etc., which behave similarly. Rule (a) says that lexical *e* may and must realize as an apostrophe only within words such as these appearing in front of a vowel or an *h*. Otherwise, it may always realize as itself (rule b).

A contraction occurs when the juxtaposition of two distinct words results in their realization as a single surface form. There are examples of both compulsory contractions (in French, preposition *à* and article *le* always contract to *au*) and of optional contractions (in

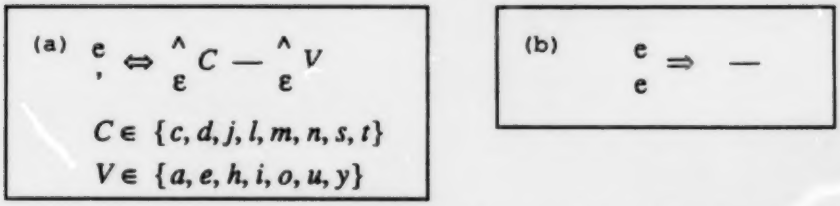


figure 9 Two-level rules controlling elisions in such French words as *ce*, *de*, *je*, etc.

English, *do* and *not* may contract to *don't*). Once again, this type of phenomena does not seem to lend itself very well to syntactic description. The set of rules in figure 10 describes the compulsory contraction of *à le* into *au*: rules (a) through (e) sanction one by one the appearance of the characters within the contraction of *à le* into *au*. Notice that in order for this to work, preposition *à* must be given a lexical form *àυ*, where υ is a lexical character which realizes as *u* within the contraction , but as ε otherwise (rule f).

Of course, the rules discussed above should always be considered as part of a larger set, which together constitutes the orthographical component. We can probably imagine starting with a set of rules describing intra-word orthography, modifying these so as to prevent them from leaping across word-boundaries (intra- and inter-word phenomena are usually distinct), and then adding on the inter-word rules (within which word-boundary markers should appear explicitly, for similar reasons). It should also be pointed out that the appearance of the lexical representation we have seen so far reflects more our desire to keep the discussion at an intuitive level than true linguistic accuracy.

It is interesting to note that while two-level rules are able to handle all three of these phenomena (word-boundary realizations, elisions and contractions), they appear to be rather ill-suited for the description of the last type (contractions). What this suggests is that although intra-word orthography *is* regular (it may be described adequately by finite-state devices), there may exist formalisms better fit to describe it than two-level rules. This idea is further supported by the fact that the set of rules required to describe the contraction of *à le*

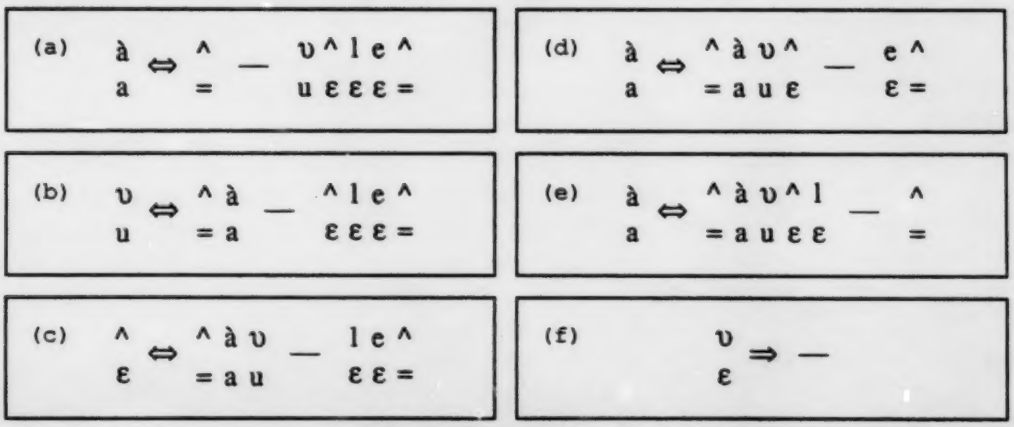


figure 10 Two-level rules in charge of the contraction of *à le* into *au*.

into *au*, it seems, could be combined in a unique and fairly simple transducer. We leave open the question of what such an alternative formalism could look like.

Another problem that appears in the course of designing orthographical rules for a language stems from the fact that some of these rules appear to be inherently ambiguous, and to generate multiple hypotheses both in analysis and in synthesis. For example, in French, surface form *des* may be analyzed either as plural form of indefinite article *un*, or as the contraction of preposition *de* and plural definite article *les*. On the other hand, the latter lexical string (juxtaposition *de+les* of the lexical forms of *de* and *les*) may produce either the contraction *des* if *les* is an article, or simply *de les* if *les* is a pronoun.

In a system such as the one we want to build, surface forms yielding multiple lexical forms are not considered a problem, because higher levels (the lexicon, morphological rules, syntactic rules, etc.) should be able to rule out unacceptable lexical forms. However, ambiguities in the other direction are problematic: for example, if the orthographical component (i.e. the component in charge of applying orthographical rules in an implementation of the model) does not 'know' if a particular instance of lexical form *les* corresponds to the article or to the pronoun, then it has no way of deciding whether a contraction with *de* must take place or not. Obviously, orthographical rules cannot work properly without this knowledge of "which rule applies in what situation". The problem, it seems, is that not all of it may reside within the orthographical component itself. If a parallel may be drawn between orthography in written text and phonology in spoken language, then the following example should be sufficient to illustrate our point. In the French phrase "*un marchand de draps anglais*", the presence of a liaison between *draps* and *anglais* depends on the scoping of adjective *anglais* (whether it applies to *marchand* or to *draps*. Well, yes, this is a bit far-fetched...). In this example, the application of a specific phonological rule is conditioned not by the immediate phonological environment, but by the underlying syntactic structure, and may even have deeper 'roots'.

So it seems reasonable to assume that not all of the knowledge relevant to orthography resides in the orthographical component itself, and that we may talk of *lexically*, *morphologically* or even *syntactically conditioned orthography*. Therefore, when orthographical ambiguities arise, the orthographical component 'knows' how to resolve it only if it is actually 'told' by some higher level component. It seems that the simplest way of modeling this transfer of knowledge in our system is through direct annotations on the lexical string. In other words, we can imagine that the lexical representation actually contains much more information than the appearance of the surface would suggest. We have already mentioned morpheme-boundary and word-boundary characters, but we could probably think of many other objects appearing in the lexical string that do have surface realizations, although not always obvious. For example, we could have the syntax (or the lexicon) to insert explicit *pauses* at certain selected positions of a sentence (or of a word) to locally inhibit some or all orthographical rules. A comma could be the surface realization of such a 'strong' pause, but 'weaker' pauses could also exist that have more subtle orthographical effects, such as inhibit the 'elision rules' in front of some words beginning with an *h*, or the 'contraction rule' of *de les* when *les* is a pronoun, etc. Other types of markers could be imagined to exist. The important thing is to see these as a way of communicating higher-level information to

the orthographical component.

Of course, lexical entries, morphological rules and even grammar rules must be written with orthography in mind, and orthographical rules must take into account the additional information, but this seems manageable. In any case, we leave open the question of how exactly this may be done.

Conclusion

The first part of this paper was a presentation of existing computational models of morphology. Many current approaches are derived from Koskenniemi's "two-level" model, so this was examined in detail first, followed by a quick survey of some of the alternative solutions proposed. The second part discussed how Koskenniemi's model could be used in a continuous-text environment to deal with "inter-word" phenomena, such as elisions and contractions.

This last topic was only briefly touched, and several questions were left open, in particular on whether the two-level model was actually the most appropriate to describe some of these phenomena. Finding a formalism better fit to this task, as well as exploring communication schemes between the orthographical component and higher levels of linguistic processing both seem to us interesting topics for future research.

In spite of all its deficiencies, we favour using Koskenniemi's model as a starting point for such research work, this for a number of practical reasons: first, there are many natural languages for the description of which the two-level formalism turns out to be quite sufficient, and to produce efficient parsers. More importantly, the model is not biased toward analysis or synthesis, and may be used for both. In our view, this is a major advantage over approaches such as those of Hankamer, Boisen, Cornell and Dolan to name but a few, which are all designed primarily to perform analyses. This is not just a convenient computational property: it is also a strong criterion of linguistic accuracy. Finally, Koskenniemi's model is, in some way, *representative* of current approaches in computational morphology. This property is important to us, because it confers a certain level of generality to our results: we believe that the ideas presented in section 5 may be applied to most finite-state morphological models.

References

[And88] Anderson, Stephen R. (1988), Morphology as a Parsing Problem, in *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dept. of Linguistics, UCLA.

[Bar86] Barton, Jr., G. Edward (1986), *The Computational Complexity of 2-level Morphology*, AI Memo 856, MIT AI Laboratory.

[Bea86] Bear, John (1986), A Morphological Recognizer with Syntactic and Phonological Rules, *COLING 86* (*Proceedings of the 11th International Conference on Computational Linguistics*), pp. 272-276.

[Bea88] Bear, John (1988), Morphology with Two-level Rules and Negative Rule Features, *COLING 88*, pp. 28-31.

[Boi88a] Boisen, Sean (1988), Parsing Morphology using Definite Clauses, in *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dept. of Linguistics, UCLA.

[Boi88b] Boisen, Sean (1988), Pro-KIMMO: a Prolog Implementation of Two-level Morphology, in *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dept. of Linguistics, UCLA.

[Cor88] Cornell, Thomas L. (1988), IceParse: A Model of Inflectional Parsing and Word Recognition for Icelandic Ablauting Verbs, in *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dept. of Linguistics, UCLA.

[DKK87] Dalrymple, Mary, Ronald M. Kaplan, Lauri Karttunen, Kimmo Koskenniemi, Sami Shaio and Michael Wescoat (1987), *Tools for Morphological Analysis*, Report No. CSLI-87-108, Stanford University CSLI.

[Dol88] Dolan, William B. (1988), A Syllable-based Parallel Processing Model for Parsing Indonesian Morphology, in *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dept. of Linguistics, UCLA.

[Han86] Hankamer, Jorge (1986), Finite-state Morphology and Left-to-right Phonology, *West Coast Conference on Formal Linguistics*, 5, pp. 41-52.

[HU79] Hopcroft, John E. and Jeffrey D. Ullman (1979), *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley.

[Kar83] Karttunen, Lauri (1983) KIMMO: A General Morphological Processor, *Texas Linguistic Forum #22*, pp. 165-186.

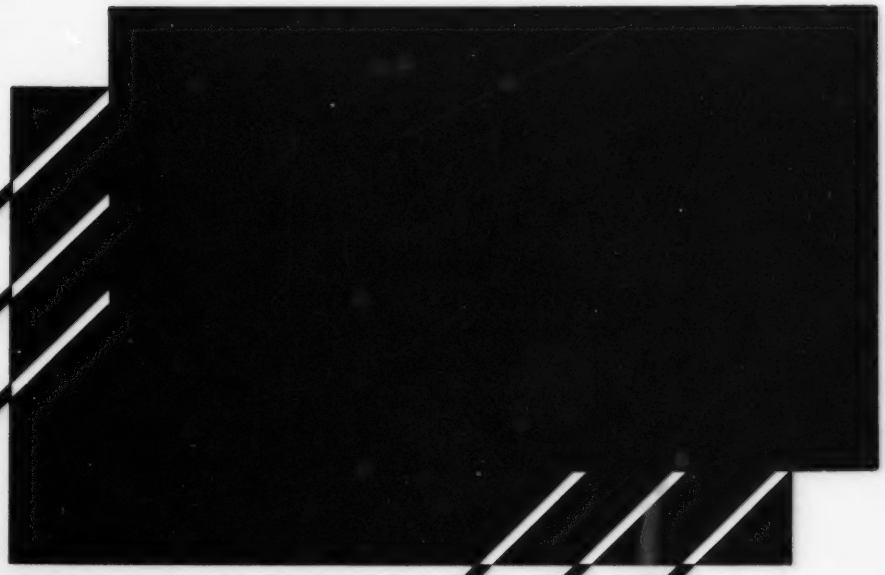
[Kos83] Koskenniemi, Kimmo (1983), *Two-level Morphology: a General Computational Model for Word-form Recognition and Production*, Publication 11, Dept. of General Linguistics, University of Helsinki.

[KW83] Karttunen, Lauri and Kent Wittenberg (1983), Two-level Morphological Analysis of English, *Texas Linguistic Forum #22*, pp. 217-228.

Gouvernement du Canada
Ministère des Communications

Government of Canada
Department of Communications

Le Centre canadien de recherche sur l'informatisation du travail
Canadian Workplace Automation Research Centre



Canada

Morphologie à états finis : Aperçu des applications et des modèles courants pour les textes continus

Michel Simard

Groupe de traduction assistée

Centre canadien de recherche
sur l'informatisation du travail

Communications Canada

Laval

mai 1990

N° de cat. Co 28-1/51-1990 F
ISBN 0-662-96175-7

Les opinions émises dans ce rapport n'engagent que l'auteur.

* This report is also available in English.

Morphologie à états finis : Aperçu des applications et des modèles courants pour les textes continus

Michel Simard[1]

Centre canadien de recherche sur l'informatisation du travail
Communications Canada
1575, boul. Chomedey
Laval (Québec)
CANADA H7V 2X2

Résumé

Les modèles morphologiques à états finis sont des formalismes servant à décrire l'ensemble des formes de mots valides d'une langue naturelle (LN). Comme ils conviennent bien à la mise en oeuvre informatique, ils ont été souvent utilisés pour créer des systèmes qui reconnaissent et génèrent efficacement des formes de mots isolées. Dans cet article, nous présentons le modèle morphologique *à deux niveaux* de Koskenniemi, que nous comparons ensuite à d'autres approches. L'intégration de tels modèles dans des systèmes de traitement de langues naturelles travaillant sur des phrases LN suppose en général que l'on modifie l'élément morphologique afin qu'il s'applique à des textes continus. Nous exposons comment ce passage des mots isolés à des textes continus peut être effectué et nous montrons qu'en plus des phénomènes orthographique normalement décrits par les modèles morphologiques, le système résultant présente des propriétés intéressantes pour la description des phénomènes 'inter-mots' tels que l'*élision* et la *contraction*.

Sujets: morphologie à deux niveaux; morphologie à états finis; linguistique informatique.

Introduction

L'intérêt à l'égard des modèles informatiques de la morphologie des langues naturelles est relativement récent. On peut le faire remonter à l'introduction du modèle 'à deux niveaux' de Kimmo Koskenniemi ([Kos83]), un modèle linguistique pour l'analyse et la synthèse morphologiques adapté à la mise en oeuvre informatique. Ce modèle consiste en un formalisme servant à décrire comment les mots d'une langue sont élaborés à partir d'un ensemble de morphèmes. On peut utiliser cette description pour générer un programme qui analyse ou synthétise des mots de cette langue. Des mises en oeuvre ont été faites en Pascal ([Kos83], en Lisp ([Kar83], [DKK87]) et en Prolog [Boi88b]) ainsi que dans d'autres

1. Email : m_simard@ccrit.doc.cdn

langages. Depuis, plusieurs systèmes informatiques de ce type ont vu le jour : ces systèmes renvoient une description d'un mot dérivée de sa décomposition en morphèmes à partir de sa 'forme de surface' (apparence physique), et vice-versa. Ils sont habituellement appelés 'modèles à états finis', car ils utilisent des dispositifs à états finis pour l'analyse. De tels formalismes conviennent très bien à la conception d'applications informatiques exigeant des descriptions lexicales de langues comportant de nombreuses flexions tels que le finnois ou le turc, ainsi que de langues "plus simples", tels que l'anglais et le français.

Un grand nombre de ces modèles à états finis ont été initialement conçus pour traiter des mots isolés. Si l'on examine la possibilité d'intégrer de tels modèles dans des milieux plus vastes de traitement de langues naturelles (p. ex., un système de traduction automatique ou une interface de base de données en langue naturelle), on en vient à se poser naturellement la question : qu'arrive-t-il lorsque des modèles morphologiques à états finis sont modifiés pour traiter des *textes continus*, comme cela se produit lorsqu'on les utilise à l'intérieur d'un système travaillant avec des *phrases* en langue naturelle? Il est intéressant de noter qu'il semble que les systèmes que l'on obtient sont capables de traiter des choses qui pourraient difficilement l'être par les modèles syntaxiques et pour lesquelles les modèles morphologiques n'ont pas été principalement conçus. Il existe divers phénomènes qui semblent dépendre de la position relative d'un mot à l'intérieur d'une phrase plutôt que de son rôle syntaxique. Ce sont par exemple les élisions et les contractions, les liaisons dans le discours ou tout simplement les 'espaces' séparant les mots dans un texte écrit. Bien que ces phénomènes soient habituellement difficiles à décrire syntaxiquement, ils entrent dans la zone de compétence de l'un des éléments du modèle à deux niveaux, celui qui traite la *phonologie* ou l'*orthographe*.

Dans la première partie de cet article, nous examinons en détail le modèle à deux niveaux de Koskenniemi : dans la section 1, nous présentons quelques concepts de base puis nous traitons le formalisme dans la section 2 et enfin envisageons sa mise en oeuvre dans la section 3. Dans la section 4, nous rappelons diverses autres approches ayant vu le jour depuis celle de Koskenniemi. La seconde partie de cet article traite la question des textes continus : dans la section 5, nous donnons une brève description de la façon dont les règles à deux niveaux de Koskenniemi peuvent être utilisées pour décrire les phénomènes intermots.

1 Éléments de base

En général, un lexique peut être considéré comme la définition de la relation entre les *formes de surface* des mots d'une langue (leur aspect physique dans le texte écrit ou dans le discours) et leurs *descriptions lexicales* (ou *définitions*). Du point de vue informatique, si les deux ensembles de formes de surface possibles et de descriptions lexicales possibles sont finis, une façon de mettre en oeuvre cette relation est de créer une liste complète des paires de forme de surface et de description lexicale satisfaisant la relation, puis d'utiliser un algorithme de recherche. Si l'un ou les deux ensembles sont infinis ou simplement très grands, cette solution devient inacceptable pour des raisons évidentes.

L'idée principale à la base du système de Koskenniemi est que cette 'relation lexica-

le' peut être décomposée et décrite complètement grâce aux descriptions de la *morphologie* et de la *phonologie* de la langue, à partir desquelles des programmes efficaces de reconnaissance et de génération peuvent être obtenus. En fait, dans le cas général, la description de ces relations individuelles est beaucoup plus compacte que la liste complète proposée ci-dessus et, bien entendu, elle est finie.

En gros, les morphèmes sont les objets à partir desquels les mots sont élaborés. Il est généralement accepté que tous les mots d'une langue sont composés d'éléments plus petits assemblés selon des règles qui définissent la *morphologie* de cette langue. Par exemple, un mot tel que *transporté* peut être considéré comme formé de trois morphèmes : le préfixe *trans-*, le radical *port* et le suffixe *-é*. On établit habituellement une distinction entre morphologie *dérivationnelle* et morphologie *inflexionnelle* : la 'dérivation' désigne les processus morphologiques selon lesquels les nouveaux mots sont obtenus à partir des anciens (p. ex., *transport* à partir de *port*), et l''inflexion' désigne les différentes transformations qu'un même et unique mot peut subir (p. ex., *transport*, *transports*)[2].

Tout ceci est rendu plus compliqué par le fait que le même morphème peut être 'réalisé' de différentes façons (c.-à-d. apparaître à la surface selon différentes formes) en fonction de son environnement, c'est-à-dire suivant la forme des morphèmes voisins. Ces variations sont contrôlées par la *phonologie* de la langue. En général, la 'phonologie' désigne ces phénomènes, tels qu'observés dans la langue parlée. Bien que l'on admette généralement que les phénomènes phonologiques du discours obéissent à des principes assez stricts, leurs équivalents dans les textes écrits (parfois appelés 'orthographe' dans la linguistique informatique) sont rarement étudiés par les linguistes, car dans la plupart des langages, les conventions orthographiques semblent plutôt arbitraires. Toutefois, afin de simplifier la suite de notre exposé, nous allons porter notre attention sur l'orthographe plutôt que sur la phonologie.

2 Le formalisme

L'idée de Koskenniemi est de "diviser le travail en deux", c.-à-d. d'introduire une représentation intermédiaire, appelée *forme lexicale*, entre la forme de surface et la description lexicale. Cette représentation résulte en fait d'une concaténation d'objets représentant les morphèmes de la langue. Lorsqu'on vérifie si une paire composée d'une forme de surface et d'une description lexicale figure dans la 'relation lexicale', on détermine s'il existe une forme lexicale satisfaisant une première relation avec la forme de surface (la relation orthographique) et une seconde relation avec la description lexicale (la relation morphologique).

La première hypothèse sous-jacente est que la description lexicale d'un mot peut être facilement calculée comme étant une fonction des descriptions des morphèmes qui le composent[3]. Il faut donc être capable de décomposer un mot en morphèmes. La seconde hypothèse est que cette décomposition est facile à condition d'avoir accès à une représentation

2. En fait, ce n'est pas aussi simple, principalement parce qu'on ne sait pas exactement ce que l'on entend par "deux mots différents" et "le même mot sous deux formes différentes", mais nous n'avons pas à nous inquiéter de ces problèmes dans cette étude.

du mot correspondant à la façon dont il se présenterait si le processus orthographique n'existait pas, c'est-à-dire si les règles orthographiques n'avaient aucun effet sur la réalisation de surface de ce mot. Ceci correspond à la représentation intermédiaire de Koskenniemi. La troisième hypothèse est que ces règles orthographiques sont faciles à appliquer et à *dé*-appliquer, ce qui fait qu'une correspondance entre la forme lexicale et la forme de surface est aussi facile à calculer. La relation morphologique est définie au moyen d'un lexique morphologique, c'est-à-dire d'un lexique de tous les morphèmes de la langue, ainsi que des règles morphologiques (quels morphèmes peuvent se combiner les uns avec les autres et de quelle façon), et la relation orthographique, avec un ensemble de règles orthographiques *à deux niveaux*. Ces deux éléments sont décrits ci-dessous de façon plus détaillée.

2.1 Le lexique

Dans le modèle à deux niveaux, le formalisme du lexique est utilisé pour définir les morphèmes et les règles morphologiques de la langue. Il se présente sous la forme d'une liste d'entrées, correspondant chacune à un morphème, et est divisé en *sous-lexiques* en fonction d'un ensemble de principes appelé la *morphotactique* de la langue (ce fractionnement des morphèmes rappelle la catégorisation des mots dans la syntaxe). On associe à chaque morphème une *forme lexicale* (son aspect dans la représentation lexicale intermédiaire), une *classe de continuation* et quelques informations connexes.

La forme lexicale d'un morphème sert en quelque sorte à désigner sa *forme canonique sous-jacente*, c.-à-d. la représentation unique à partir de laquelle toutes ses réalisations de surface sont dérivées conformément aux règles orthographiques de la langue. Il revient à celui qui produit un lexique de choisir l'aspect de ces formes lexicales, mais, dans la plupart des cas, elles ne devraient différer que de façon mineure de leurs équivalents de surface : une forme lexicale devrait être quelque chose comme une *généralisation* de toutes les réalisations de surface du morphème auquel elle se rattache. Donc, l'*alphabet lexical* d'une langue, c.-à-d. l'alphabet dans lequel ces formes sont écrites, bien que pouvant être n'importe quoi, renfermera probablement tous les caractères de l'alphabet de surface avec peut-être en plus un petit ensemble de caractères supplémentaires (diacritique, etc.). Par exemple, la forme lexicale du verbe anglais *to carry* pourrait être quelque chose comme *carrÿ*, dans lequel le caractère ÿ désigne une généralisation des réalisations de surface *y* (comme dans *carry*) et *i* (comme dans *carried*). Une autre possibilité consiste à tout simplement utiliser l'une des réalisations de surface : dans ce cas, *carry* serait aussi une forme lexicale acceptable du verbe *to carry*, à condition que le *y* et le *i* soient tous deux considérés comme des réalisations de surface possibles du caractère lexical *y*.

Chaque *classe de continuation* renvoie à l'ensemble des sous-lexiques renfermant les morphèmes pouvant être concaténés à la droite du morphème courant : ces classes constituent le moyen selon lequel les *règles morphologiques* d'une langue sont définies dans ce

---

3. lorsqu'on utilise l'expression "facile à calculer", on parle de la notion intuitive d''efficacité pratique' plutôt que de celle de 'résolubilité'. Bien que le problème général de l'analyse des langues défini par le modèle de Koskenniemi se soit révélé comme étant complet-NP (voir [Bar86]), dans la plupart des cas réels, des analyseurs efficaces peuvent être élaborés.

modèle. Certains sous-lexiques sont identifiés comme étant des lexiques de *radicaux*. Ils renferment des morphèmes pouvant 'commencer' les mots de la langue. Enfin, il y a aussi une classe de continuation *finale* spéciale, qui renvoie à un ensemble vide de sous-lexiques et qui est affectée aux morphèmes pouvant 'terminer' un mot. Les formes lexicales des mots d'une langue peuvent alors être obtenues par concaténation des morphèmes du lexique selon les façons prescrites par les classes de continuation, en commençant par un morphème 'radical' et en terminant par un morphème de la classe de continuation 'finale'.

La figure 1 illustre un tel lexique pour un (très) petit sous-ensemble de l'anglais. Les rectangles renferment les sous-lexiques, et les éllipses, les classes de continuation, les morphèmes sont en italiques (en fait il s'agit des formes lexicales de ces morphèmes), et *Vrad* et *Nrad* sont les sous-lexiques de radicaux. Les morphèmes qui ne pointent pas vers une classe de continuation donnée sont considérés comme étant 'finaux'. Ce lexique permet d'obtenir des mots tels que :

mice *eat0* *cry+ed* *walk+er+s+'s*

Vous remarquerez qu'en pratique, le mot 'morphème' peut être pris dans un sens très vague lorsqu'on élabore un lexique. Par exemple, s'il est commode dans un certain contexte de

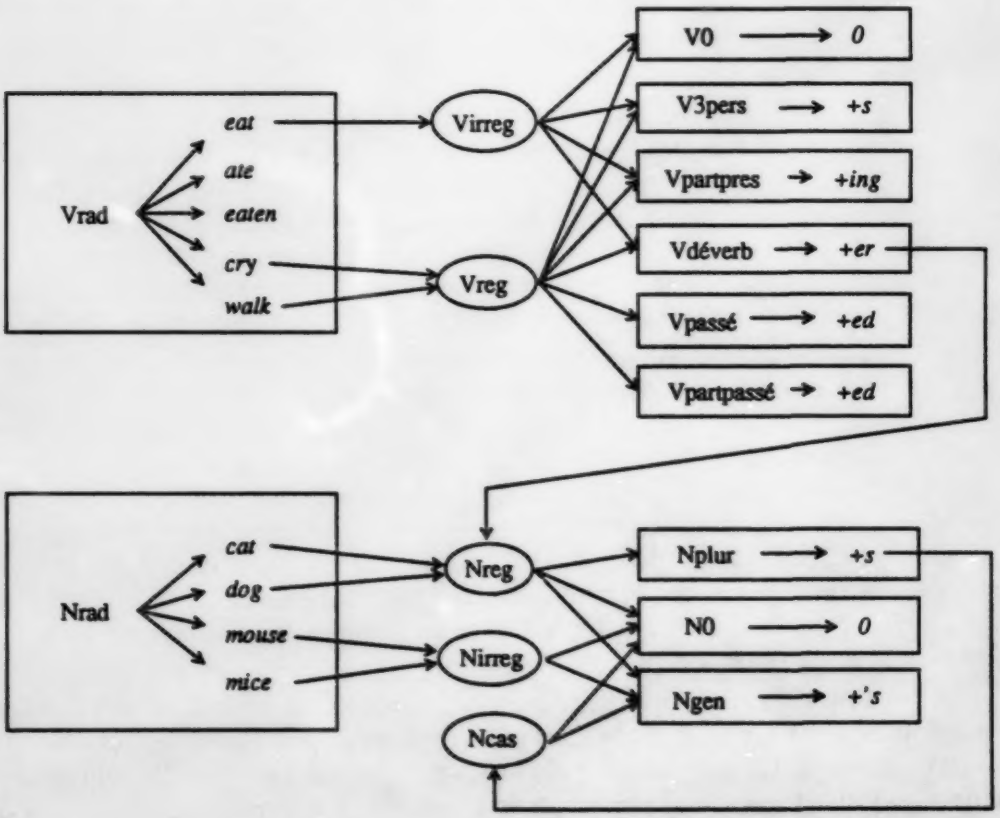


figure 1 Exemple de lexique anglais.

considérer l'expression *danse à claquette* comme un 'morphème', rien n'empêche de le faire. D'un point de vue purement linguistique, cela peut paraître quelque peu choquant, mais, dans la pratique, c'est tout à fait courant.

Pour des raisons de clarté, nous avons omis l'information liée aux morphèmes, mais, dans le modèle original, elle se présente sous la forme d'une chaîne de texte. La description lexicale d'un mot est le résultat de la concaténation des descriptions individuelles de ses morphèmes dans leur ordre d'occurrence. Par exemple, si le morphème *cry* est relié à l'information "*verbe 'to cry'*" et le suffixe *+ed*, à "*passé*", la forme lexicale *cry+ed* correspond à la description lexicale "*verbe 'to cry' passé*". Cette façon plutôt primitive de rassembler des informations lexicales constitue un expédient théorique dont le seul but est de simplifier la démonstration de l'exactitude des analyses de Koskenniemi. Dans les mises en oeuvre ultérieures de ce modèle, le format des descriptions lexicales ainsi que l'effet de la description d'un morphème sur celles des mots dans lesquels il figure doivent être définis par l'utilisateur.

2.2 Règles à deux niveaux

Les règles à deux niveaux définissent les relations qui existent entre les formes lexicales et les formes de surface. Elles spécifient des correspondances *caractère-à-caractère* entre les deux représentations. Ainsi, elles définissent les règles orthographiques de la langue. Elles ont le format suivant :

<pc> <opérateur> <cg> — <cd>

où *<opérateur>* est l'un des opérateurs ⇒, ⇐ et ⇔, *<pc>* est un *ensemble de paires concrètes* (ou *EPC*), et *<cg>* et *<cd>* sont des *expressions régulières de paires* (ou *ERP*), c.-à-d. des expressions régulières définies sur un alphabet de EPC. Ces EPC sont des paires *(x,y)*, où *x* et *y* sont des éléments de l'alphabet lexical et de l'alphabet de surface, respectivement, qui décrivent des réalisations possibles des caractères lexicaux à la surface. Donc, *(x,y)* peut être considéré comme le "caractère lexical *x* et sa réalisation de surface *y*".

Certains caractères ont une signification spéciale à l'intérieur des EPC : le signe = signifie 'n'importe quoi'; par exemple, *(=,s)* représente "n'importe quel caractère lexical et sa réalisation de surface *s*". Réciproquement *(s,=)* désigne le "caractère lexical *s* et sa réalisation de surface, quelle qu'elle soit". Le caractère ε désigne le caractère 'nul' : *(s,ε)* signifie le "caractère lexical s, qui est réalisé par la chaîne 'nulle' à la surface". Nous utilisons parfois des variables : les lettres majuscules sont normalement employées pour les désigner, comme par exemple dans *(V,x)*, $V \in \{a,e,i,o,u\}$. Enfin, si aucune confusion n'est possible, un caractère unique *x* sert à désigner une paire *(x,x)* dans laquelle le caractère lexical est réalisé par lui-même à la surface (il existe d'autres raccourcis notationnels, mais nous nous limitons à ceux-ci pour l'instant).

Le *<pc>* est la *partie correspondance* de la règle à deux niveaux, c'est-à-dire la partie qui spécifie la paire visée par la règle, et *<cg>* et *<cd>* sont le *contexte de gauche* et le *contexte de droite*, respectivement, c'est-à-dire les parties qui spécifient le contexte à l'intérieur duquel la règle s'applique. La partie *<opérateur>* spécifie le *type* de la règle de la façon décrite ci-dessous.

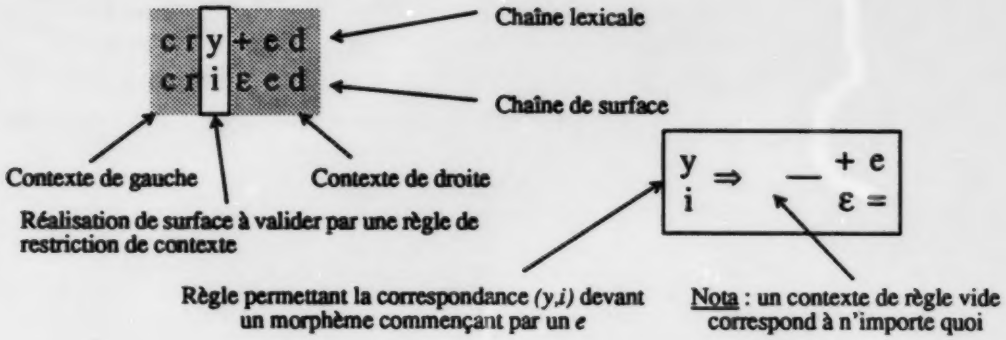


figure 2 Paire de formes lexicale et de surface correspondantes et règle de restriction de contexte associée.

Les *règles de restriction de contexte* prennent la forme

<*pc*> ⇒ <*cg*> — <*cd*>

Elles spécifient un contexte <*cg*> — <*cd*> à l'intérieur duquel la correspondance <*pc*> *peut* s'effectuer. Autrement dit, si <*pc*> est *(x,y)* et que *x* figure dans une forme lexicale entouré de <*cg*> à gauche et de <*cd*> à droite, il peut alors être réalisé par *y* à la surface (nous disons qu'une paire *(x,y)* est entourée de <*cg*> et de <*cd*> si ce qui apparaît immédiatement à gauche et à droite de *(x,y)* correspond aux expressions régulières <*cg*> et <*cd*>, respectivement). En général, pour qu'une paire composée d'une forme lexicale et d'une forme de surface soit orthographiquement correcte, chacune des paires correspondantes de caractère lexical et de caractère de surface doit être validée par une règle de restriction de contexte. Ceci est illustré à la figure 2. On remarquera que les EPC sont écrits verticalement, avec le caractère lexical en haut. Il s'agit de la notation standard (mais cela ne s'adapte pas bien à un texte écrit et pour cette raison nous alternons entre les deux notations).

Les *règles de coercition de surface* prennent la forme

<*pc*> ⇐ <*cg*> — <*cd*>

Elles spécifient le contexte à l'intérieur duquel une correspondance <*pc*> *doit* s'effectuer : si <*pc*> est *(x,y)* et que *x* figure à l'intérieur d'une chaîne lexicale entourée de <*cg*> et de <*cd*>, il doit alors être réalisé par *y* à la surface. Pour qu'une paire composée d'une forme lexicale et d'une forme de surface soit orthographiquement correcte, aucune de ses paires de caractères individuelles ne doit contredire l'une quelconque des règles de coercition de surface. La figure 3 illustre ce que l'on entend par une 'contradiction'. On remarquera que l'existence d'une règle de coercition de surface ayant pour but d'obliger une certaine correspondance *(x,y)* ne suppose pas implicitement que cette correspondance est *permise*. Une telle règle spécifie quelque chose comme "Quelle que soit la réalisation normalement permise de *x*, dans ce contexte, il ne doit pas être réalisé par autre chose que *y*". Pour cette raison, ce type de règle s'avère surtout utile pour spécifier des correspondances *interdites*,

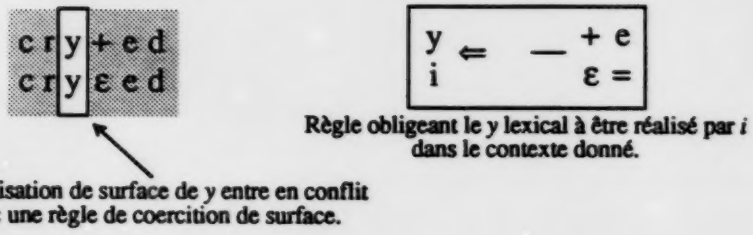


figure 3 Exemple d'une paire lexicale/de surface régie par une règle de coercition de surface.

comme par exemple dans :

$$\begin{matrix} a \\ -b \end{matrix} \Leftarrow c — d$$

qui signifie "entouré de *c* et de *d*, le caractère *a* doit être réalisé par un caractère qui n'est pas égal à *b* (*a* ne peut pas être réalisé par *b*)". Donc, une telle règle *empêche* qu'une correspondance donnée s'effectue. Il s'agit alors de vérifier les règles de restriction de contexte correspondantes pour déterminer en quoi *a* peut être réalisé dans le contexte.

Les *règles composites* sont utilisées pour spécifier des correspondances qui sont à la fois permises et obligatoires. On les obtient en combinant une règle de restriction de contexte et une règle de coercition de surface qui ont la même partie correspondance et les mêmes contextes de droite et de gauche :

<*pc*> ⇔ <*cg*> — <*cd*> est équivalent à

<*pc*> ⇒ <*cg*> — <*cd*> et

<*pc*> ⇐ <*cg*> — <*cd*>.

Des descriptions complètes de l'orthographe d'une langue sont, en principe, des ensembles de règles à deux niveaux. Pour une paire de caractères donnée, toutes les règles de restriction de contexte sont prises *disjonctivement* (l'une d'entre elles doit être satisfaite, c.-à-d. correspondre au caractère courant et au contexte), et cette disjonction est ensuite prise *conjonctivement* avec toutes les règles de coercition de surface (toutes ces règles doivent être satisfaites, c.-à-d. être sans contradiction si elles correspondent au contexte courant et au caractère lexical). Pour qu'une paire composée d'une forme de surface et d'une forme lexicale soit acceptée par l'ensemble des règles, cet énoncé logique doit être satisfait à chaque position de la chaîne. Comme les règles énoncent simplement des relations entre les deux niveaux de représentation, le formalisme se prête indifféremment à l'analyse ou à la synthèse.

3 La mise en oeuvre

L'idée principale sur laquelle repose la mise en oeuvre du modèle de Koskenniemi est que ses deux formalismes (lexique et règles à deux niveaux) définissent des langages réguliers pour lesquels des analyseurs efficaces peuvent facilement être élaborés. Des dispositifs à états finis individuels correspondant au lexique et aux règles à deux niveaux peuvent être

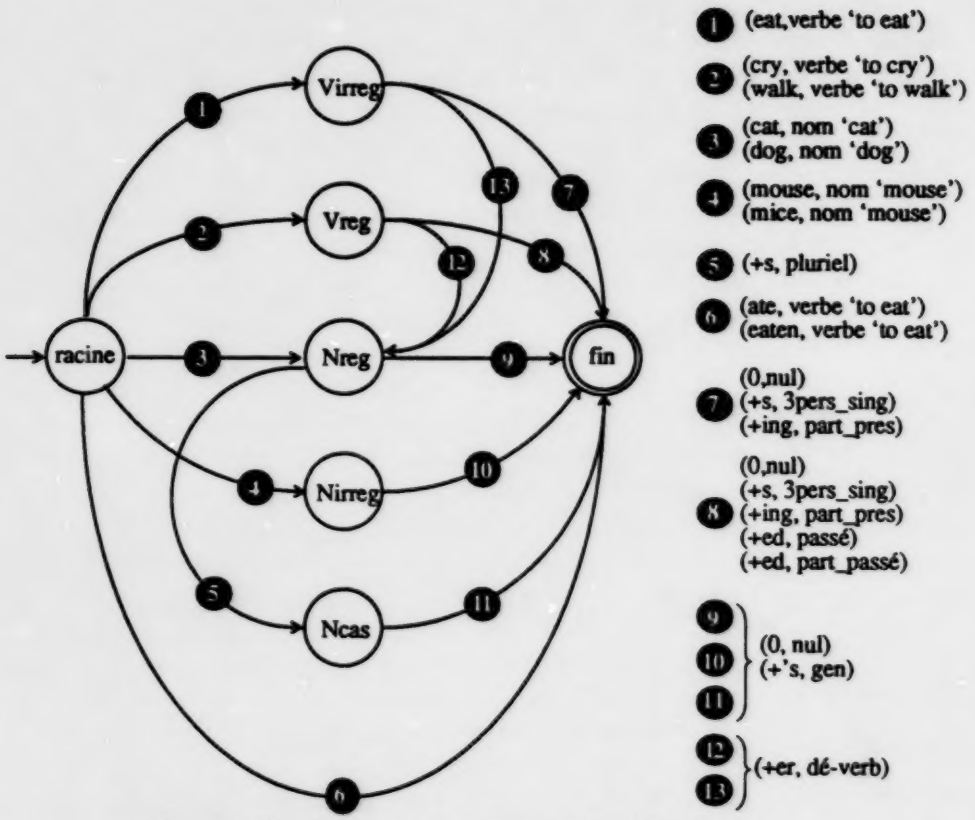


figure 4 Transducteur correspondant au lexique de la figure 1

produits puis utilisés en parallèle pour analyser ou générer des formes de surface. La méthode utilisée est exposée ci-dessous.

Un *transducteur à états finis* est un type particulier d'automate à états finis qui reconnaît une langue définie sur un alphabet de paires de caractères. Formellement, si A et B sont des alphabets, un automate qui reconnaît une langue définie sur $(A \times B)^*$ est un transducteur. Ce type d'appareil de calcul nous intéresse, car il permet de décrire les relations entre des chaînes définies sur différents alphabets, à condition que le calcul de ces relations n'exige qu'une quantité finie de mémoire. En fait, il semble que c'est le cas des relations qui existent entre les descriptions lexicales et les formes lexicales, et entre les formes lexicales et les formes de surface.

Soit un lexique donné conforme au formalisme de Koskenniemi, considérons que l'ensemble de tous les morphèmes forme un alphabet (c.-à-d. chaque morphème du lexique est un 'symbole' dans l'alphabet). Il est alors évident que les règles morphologiques, telles qu'elles sont codées au moyen des classes de continuation, définissent un langage régulier s'appliquant à cet alphabet. De même, si les descriptions lexicales des mots sont obtenues par concaténation des descriptions des morphèmes individuels, les règles morphologiques définissent aussi un langage régulier sur l'alphabet des descriptions associées aux morphè-

mes. Il est ensuite facile de voir comment il est possible d'élaborer un transducteur qui reconnaît de façon précise toutes les paires de descriptions et de morphèmes bien formées lexicalement. Un dispositif correspondant au lexique de la figure 1 est illustré à la figure 4 (nous avons utilisé dans cet article les conventions de représentation de [HU79] pour les dispositifs à états finis). Il devient alors trivial de transformer ce transducteur en un autre qui traite des formes lexicales plutôt que des morphèmes (voir figure 5), de façon à ce que nous puissions obtenir un dispositif de calcul efficace, capable de reconnaître des paires bien formées de descriptions lexicales et de formes lexicales.

Bien entendu, nous voulons faire bien plus que de simplement reconnaître des paires bien formées : nous voulons aussi effectuer des opérations d'*analyse* et de *synthèse*. Formellement, soit un transducteur M reconnaissant un langage $L(M)$ défini sur $(A \times B)^*$, une analyse d'une chaîne s de A^* est une chaîne t de B^* telle qu'il existe une chaîne x de $L(M)$ avec une projection sur A $\Pi_A(x) = s$ et une projection sur B $\Pi_B(x) = t$ (la synthèse est définie d'une manière symétrique). Des méthodes efficaces de recherche existent pour calculer de telles fonctions au moyen de transducteurs, mais nous ne les traiterons pas dans cet article.

Des règles orthographiques à deux niveaux définissent une langue sur un alphabet de paires composées chacune d'un caractère lexical et d'un caractère de surface. Ces règles peuvent être reconnues par des transducteurs dont l'élaboration n'est pas aussi évidente que dans le cas des lexiques. En fait, la méthode de production de ces dispositifs à partir des ensembles de règles à deux niveaux est complexe, et, à notre connaissance, il n'en existe aucune description claire et formelle. Essentiellement, un ensemble de règles à deux niveaux peut être divisé en des sous-ensembles de règles partageant la même partie correspondance (partie <*pc*>), et un transducteur peut être élaboré pour chacun de ces sous-ensembles de façon qu'une chaîne définie sur les paires de caractères lexicaux et de caractères de surface soit acceptée comme étant orthographiquement exacte si et seulement si elle est reconnue par tous les transducteurs.

La figure 6 illustre un tel transducteur correspondant à la règle composite responsable de la correspondance (*y*,*i*) en anglais (les états d'acceptation sont entourés d'un cercle double). Dans cet exemple, le symbole = à l'intérieur d'une paire de caractères désigne 'n'importe quoi' (comme auparavant), et le symbole ~ désigne une 'négation'. Par exemple, la paire (*y*, ~*i*) désigne le caractère lexical *y* réalisé à la surface par n'importe quoi sauf *i*, ~(*y*,*i*) désigne toutes les paires différentes de (*y*,*i*), et ~(*y*,=) correspond à n'importe quelle

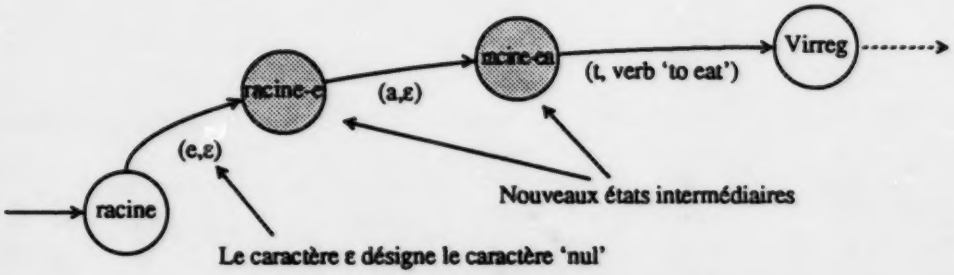


figure 5 Expansion d'une transition du transducteur lexical.

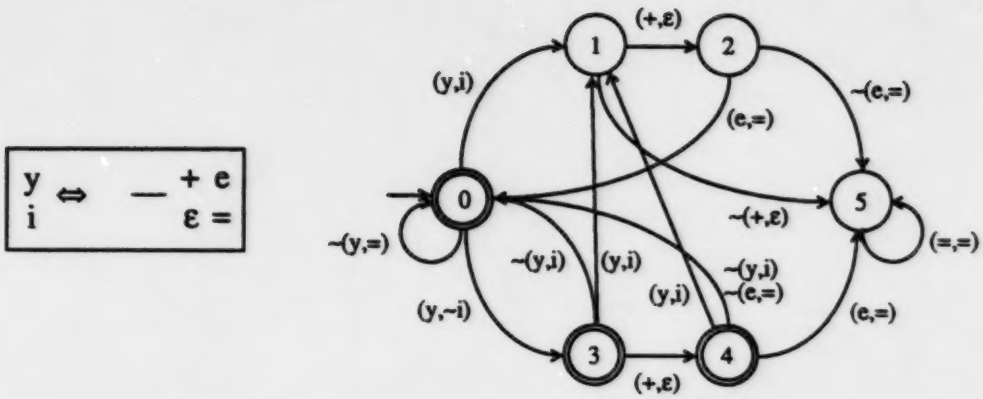


figure 6 Règle composite à deux niveaux et transducteur correspondant.

paire de caractères dont le caractère lexical n'est pas *y*, quel que soit le caractère de surface. Le transducteur est élaboré de façon que, lorsqu'on commence à l'état 0, n'importe quelle chaîne soit acceptée, à l'exception de celles renfermant une occurrence de (*y*,*i*) qui n'est pas immédiatement suivie de (+,*e*) et (*e*,=), et de celles renfermant un caractère lexical *y* qui n'est pas réalisé par *y* et qui est immédiatement suivi de (+,*e*) et (*e*,=). Ceci reflète l'objectif de la règle, qui est que *y doit* être réalisé par *i* dans le contexte donné, et que, comme il s'agit de la seule règle concernant la correspondance (*y*,*i*), c'est aussi le seul contexte à l'intérieur duquel il *peut* figurer. S'il existe une autre règle concernant cette correspondance, *les deux* règles doivent alors être mises en oeuvre à l'intérieur du même transducteur.

En fait, comme les ensembles réguliers sont fermés sous l'intersection, il est possible de combiner tous les transducteurs obtenus à partir d'un ensemble de règles à deux niveaux pour en former un seul. En théorie, cela produit un dispositif à états finis dont le nombre d'états est un multiple du nombre des états des éléments individuels à partir desquels il a été obtenu. Toutefois, la nature du 'problème orthographique' est telle que le transducteur résultant peut habituellement être optimisé afin que le nombre d'états soit plus proche de la somme que du produit, ce qui est intéressant lorsqu'on attache de l'importance à l'efficacité.

Enfin, si nous combinons des transducteurs de la façon décrite ci-dessus, nous obtenons au bout du compte seulement deux transducteurs : un qui définit un langage sur des paires de caractères de surface et de caractères lexicaux (le 'langage orthographique') et l'autre sur des paires de descriptions lexicales et de formes lexicales (le 'langage morphologique'). La façon dont ces dispositifs interagissent les uns avec les autres dépend des mises en oeuvre, mais nous pouvons imaginer qu'il existerait une sorte de relation 'consommateur-producteur' entre les deux (figure 7); autrement dit, l'un d'entre eux 'consomme' la chaîne lexicale à mesure qu'elle est 'produite' par l'autre. Les identités du 'producteur' et du 'consommateur' sont déterminées par le fait que le système fonctionne en mode d'analyse ou de synthèse. Le système résultant est un analyseur gauche à droite qui, suivant bien entendu la nature exacte du lexique et des règles morphologiques et orthographiques,

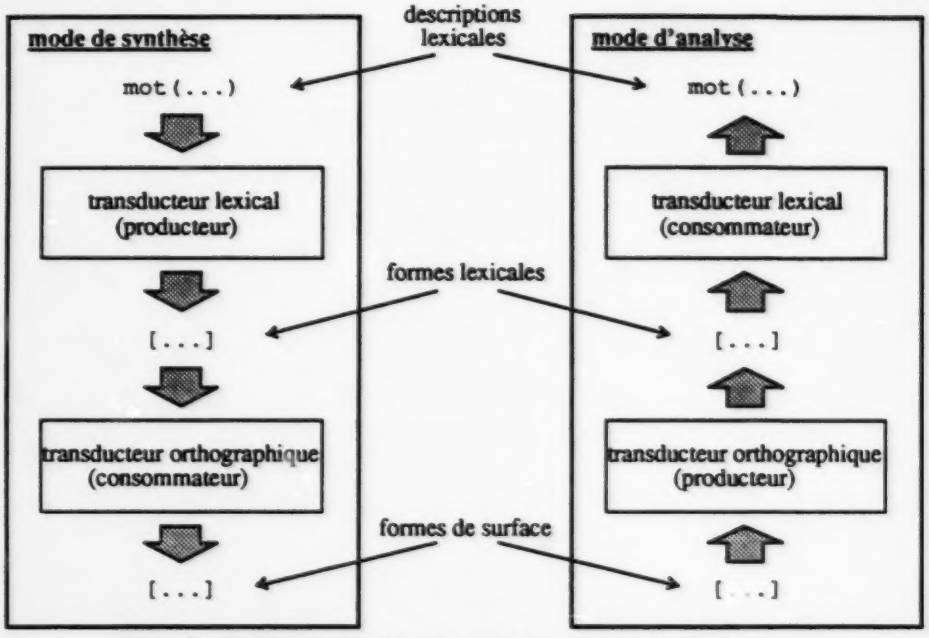


figure 7 Relation du type 'producteur-consommateur' entre les transducteurs orthographique et lexical.

peut être très efficace tout aussi bien dans l'analyse que dans la synthèse.

4 Autres Approches

Le modèle à deux niveaux de Koskenniemi étant l'un des quelques modèles de calcul pour la morphologie et l'orthographe qui présentent un certain intérêt pour un linguiste et étant aussi le plus connu, il a été l'objet de diverses critiques. Examinons maintenant les principales différences d'opinion ainsi que certaines des autres solutions proposées.

- *Codage des règles morphologiques* : Le modèle à deux niveaux est souvent critiqué à cause de la 'faiblesse' de ses règles morphologiques : bien qu'on ne puisse pas prouver qu'il est insuffisant, il semble trop souvent inadéquat pour décrire les phénomènes morphologiques autres que la suffixation et la préfixation. Dolan ([Dol88]) énumère des notions telles que la *reduplication* (répétion d'un morphème à l'intérieur d'un mot), l'*infixation* (l'insertion d'un morphème 'à l'intérieur' d'un autre) et la *circonfixation* (morphème entouré d'un autre), tandis qu'Anderson ([And88]) et Boisen ([Boi88a]) parlent de phénomènes 'non affixaux' (p. ex., l'alternance vocalique dans certains verbes anglais : *sing*, *sang*, *sung*) comme d'autres sources de difficultés. Normalement, le codage de ces phénomènes à l'intérieur du formalisme à deux niveaux exige beaucoup de lexicalisation (p. ex., il faut entrer toutes les formes du verbe *to sing* en tant que morphèmes distincts dans le lexique) et de duplication de l'information (p. ex., l'indication des combinaisons facultatives d'éléments de certains sous-lexiques au moyen d'un 'circonfixe' exige la duplication du sous-lexique au complet).

Ceci a été amélioré dans des présentations et des mises en oeuvre ultérieures du modèle. En particulier, des méthodes plus sophistiquées de codage et d'élaboration des descriptions lexicales (structures d'attributs, etc.; voir [DKK87]) ont été élaborées, mais de nombreux chercheurs ont suggéré que le problème découle principalement du fait que les classes de continuation forme un 'squelette' régulier et ils ont choisi un formalisme de règles de 'structures de phrases', une idée qui a été premièrement suggérée par Karttunen et Wittenburg [KW83]. Par exemple, Bear décrit les règles morphologiques en utilisant un formalisme de type PATR ([Bea86]), et le formalisme de Boisen est fondé sur les DCG ([Boi88a]).

• *Application parallèle des règles orthographiques* : Les modèles traditionnels de phonologie générative décrivent les phénomènes phonologiques en utilisant des règles de réécriture, destinées à être appliquées 'en cascade'. Autrement dit, chaque règle s'applique à la sortie de la règle précédente, et sa sortie à la suivante. Les correspondances entre la forme de surface et la forme lexicale sous-jacente s'obtiennent en passant à travers plusieurs couches de représentations intermédiaires, correspondant à la situation entre les applications des règles. Une hypothèse sur laquelle est fondé le modèle de Koskenniemi est que tous les niveaux intermédiaires entre la forme lexicale et la forme de surface peuvent être contournés, et que les correspondances entre les deux niveaux peuvent être décrites directement avec des règles qui s'appliquent simultanément.

Cette opinion n'est pas partagée par tous, et certains modèles continuent de préférer l'ancienne méthode. Par exemple, Hankamer propose une méthode d'analyse du turc qui, bien qu'étant très proche de celle de Koskenniemi en ce qui concerne le codage des règles morphologiques, préconise une application en cascade des règles orthographiques (voir [Han86]).

• *Séparation entre l'orthographe et la morphologie* : L'un des plus importants principes sur lequel est fondé le modèle de Koskenniemi est la séparation présumée entre le traitement orthographique et le traitement morphologique : tout ce que les règles orthographiques doivent connaître au sujet des niveaux plus élevés (morphologie, syntaxe, etc.) est transmis à travers la représentation lexicale intermédiaire, et vice-versa. Cette opinion est souvent contestée ainsi qu'on peut le voir dans les exemples suivants.

Dans un modèle qui autrement est très proche de celui de Koskenniemi, Bear ([Bea88]) propose de relier ce qu'il appelle des 'attributs de règles négatives' aux morphèmes du lexique. Ces attributs ont pour objectif d''invalider localement' certaines règles orthographiques, afin qu'elles ne soient pas appliquées lorsqu'il se produit des 'exceptions'. Cela signifie que l'élément morphologique a un contrôle direct sur le comportement de l'élément orthographique. On remarquera que ceci ne produit pas un formalisme plus puissant que celui de Koskenniemi : l'emploi d'attributs de règles négatives est absolument équivalent à l'utilisation d'annotations dans la chaîne lexicale (voir section 5). Toutefois, cette méthode présente l'avantage qu'il n'est pas nécessaire de tenir compte des exceptions lorsqu'on écrit des règles orthographiques mais seulement lorsqu'on rédige les entrées lexicales auxquelles ces exceptions s'appliquent.

Dans le même ordre d'idée, Hankamer ([Han86]) utilise des ensembles différents de règles pour les *radicaux* et pour les *suffixes*, et suppose ainsi que les règles orthographiques 'connaissent' le type de morphème qu'elles traitent. En fait, il s'agit simplement d'une autre méthode plus restreinte que celle de Bear, destinée à invalider localement des règles sélectionnées.

Le modèle d'analyse inflectionnelle 'IceParse' de Cornell ([Cor88]) se distingue plus nettement de l'approche de Koskenniemi : il considère la morphologie comme une sorte de 'phénomène orthographique' (par exemple dans les relations d'umlaut et d'alternance vocalique). Donc, dans ce modèle, il n'existe aucune entrée lexicale correspondant aux suffixes, les règles morphologiques sont indiquées de la même façon que les règles orthographiques et elles définissent des relations entre les mêmes deux niveaux de représentation, c'est-à-dire la forme de surface et une certaine représentation sous-jacente apparentée à la forme lexicale de Koskenniemi. La différence entre les deux types de règles est que les règles morphologiques sont capables d'avoir un effet sur les descriptions lexicales des lexèmes auxquels elles s'appliquent. Au niveau de la mise en oeuvre, les deux types de règles sont traduites en des transducteurs à états finis.

Tout à l'opposé, dans le formaliste DCM de Boisen, les règles orthographiques sont codées sous forme de règles transformationnelles et appliquées 'à l'intérieur' des règles morphologiques. Autrement dit, selon Boisen, l'application d'une règle orthographique donnée est toujours une conséquence de l'application d'une règle morphologique spécifique.

5 Utilisation du formalisme de Koskenniemi avec des textes continus

Comme nous l'avons mentionné ci-dessus, lorsqu'on envisage l'intégration d'un formalisme tel que celui de Koskenniemi dans un système de traitement de langue naturelle, la situation suivante se présente : bien que le modèle à deux niveaux ait été initialement conçu pour traiter des mots simples isolés, nous voulons qu'il fonctionne avec des textes continus, un environnement dans lequel les séparations entre les mots ne sont pas toujours clairement indiquées. Par exemple, bien que dans la plupart des langues écrites, le caractère normal de délimitation entre les mots est l''espace', il est probable que des termes tels que *Colombie Britannique* et *ministère de l'Agriculture* seront considérés comme des mots ne pouvant en aucun cas être décomposés. Cela signifie qu'en plus de reconnaître des mots, nous devons aussi être capables de les isoler dans le texte.

Dans le cas de mots uniques, un programme mettant en oeuvre le modèle à deux niveaux doit normalement répondre à des demandes du type "Donner une description lexicale du mot suivant s'il en existe une" (la 'demande d'analyse') ou "Produire si cela est possible un mot ayant cette description lexicale" (la 'demande de synthèse'). Il nous semble tout à fait évident qu'un programme capable de répondre à ces demandes, auquel on fournirait seulement quelques informations supplémentaires, pourrait être utilisé pour répondre à des demandes au sujet de *phrases*, telles que "Trouver tous les mots de la langue qui correspondent au côté gauche de cette phrase ainsi que leurs descriptions lexicales" ou "Produire un

mot ayant cette description lexicale à cet emplacement de la phrase". Dans ce cas, le mot 'phrase' désigne simplement une 'liste de mots' et les "quelques informations supplémentaires" sont en fait une description de la façon dont les mots figurent à l'intérieur des phrases (aspect des délimiteurs des mots, etc.).

À notre avis, on peut coder cette information sur les phrases sous forme *morphologique* et *orthographique* en introduisant un 'caractère explicite de délimitation de mot' à la fin de chaque mot de la langue – chose triviale dans l'encodage des règles morphologiques à l'aide du formalisme à deux niveaux, à condition que ce caractère soit traité en tant que morphème – et en contrôlant 'orthographiquement' sa réalisation de surface avec des règles à deux niveaux. Sans entrer trop avant dans les détails, nous pouvons voir comment procéder. Supposons que le symbole '^' est le caractère lexical de délimitation de mot. En général, il sera réalisé à la surface par un espace (que nous représenterons '_' dans ce qui suit). Toutefois, il existe diverses situations dans lesquelles aucun espace ne doit apparaître, soit par exemple avant un caractère de ponctuation ou après une apostrophe. On peut alors utiliser un ensemble de règles à deux niveaux (voir la figure 8). La règle (a) indique que le '^' peut être réalisé par un espace '_' à la surface . Toutefois, d'autres règles spécifient des situations dans lesquelles il peut et doit être réalisé par le caractère nul ε : avant un caractère de ponctuation (règle b) ou après une apostrophe (règle c).

On remarque que ceci ne fonctionnera que si les règles orthographiques sont appliquées à des phrases complètes et non pas à des mots individuels. En général, la réalisation d'un délimiteur de mot dépendra de l'aspect de ce qui précède et de ce qui suit. Dans l'intégration du formalisme à deux niveaux dans un système de traitement de langue naturelle, l'un de nos principaux soucis sera de permettre cette application 'globale' des règles.

En outre, il est intéressant de noter qu'il semble qu'en plus de l'orthographe 'intra1-mots' standard, l'aspect de surface des délimiteurs de mot n'est pas la seule chose pouvant être traitée orthographiquement lorsqu'on applique des règles à deux niveaux de cette façon. Par exemple, les *élisions* et les *contractions* peuvent également être considérées comme des 'phénomènes orthographiques inter-mots'. Jetons un coup d'oeil rapide sur cette question : en français, un grand nombre de mots prennent une forme différente suivant qu'ils apparaissent devant une voyelle (ou un *h* muet) ou une consonne. Par exemple, le *e*

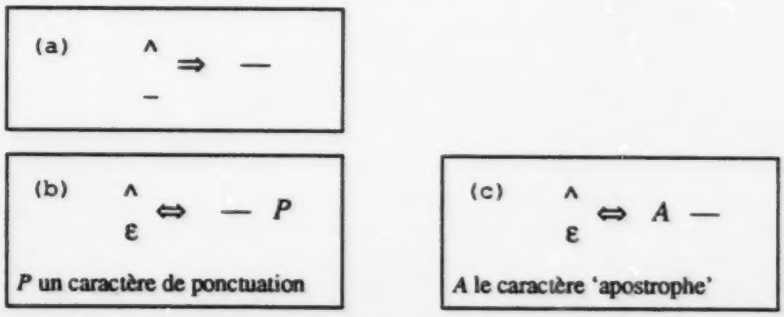


figure 8 Règles à deux niveaux régissant les réalisations du caractère de délimitation de mot '^'

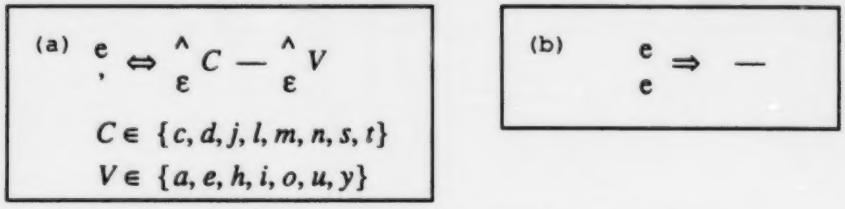


figure 9 Règles à deux niveaux régissant l'élision dans des mots français tels que *ce*, *de*, *je*, etc.

du mot *le* (article défini ou pronom) disparaît et est remplacé par une apostrophe (') avant une voyelle, comme dans *l'avion* (il se produit quelque chose de semblable en anglais avec l'article indéfini *a*). Ce phénomène est difficile à décrire syntaxiquement, car il ne semble dépendre en aucune façon de la relation syntaxique entre le mot élidé (dans ce cas, *le*) et le mot ayant provoqué l'élision.

La règle à deux niveaux de la figure 9 résoud ce problème, non seulement pour le *le* mais aussi pour bien d'autres mots qui se comportent de la même façon tels que *ne*, *se*, *me*, *te*, etc. La règle (a) indique que le *e* lexical peut et doit être réalisé par une apostrophe seulement à l'intérieur de mots tels que ceux-ci lorsqu'ils apparaissent devant une voyelle ou un *h*. Autrement, il est toujours réalisé par lui-même (règle b).

Il y a contraction lorsque la juxtaposition de deux mots distincts donne une réalisation ayant une forme de surface simple. Il existe des exemples de contractions obligatoires (en français, la préposition *à* et l'article *le* deviennent *au* en se contractant) et de contractions facultatives (en anglais, *do* et *not* peuvent se contracter en *don't*). De nouveau, ce type de phénomène ne semble pas se prêter très bien à une description syntaxique. L'ensemble des règles données à la figure 10 décrit la contraction obligatoire de *à le* en *au* : les règles (a) à (e) sanctionnent un par un l'aspect des caractères à l'intérieur de la contraction de *à le* en *au*. On remarquera que pour que ceci fonctionne, il faut affecter à la préposition *à* la forme lexicale *àυ* , où υ est un caractère lexical qui se réalise en *u* dans la contraction, et en ε autrement (règle f).

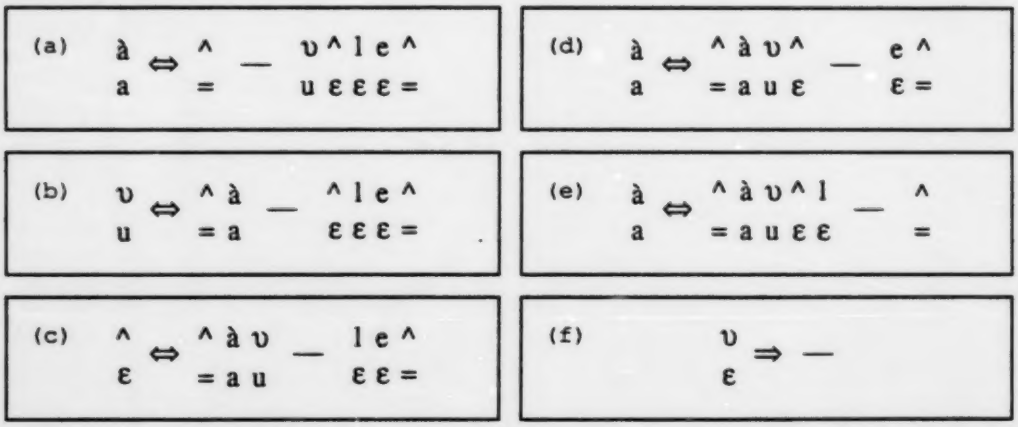


figure 10 Règles à deux niveaux régissant la contraction de *à le* en *au*.

Bien entendu, les règles abordées ci-dessus doivent toujours être considérées comme faisant partie d'un ensemble plus grand, qui forme l'élément orthographique. Nous pouvons imaginer de commencer avec un ensemble de règles décrivant l'orthographe intra-mots, puis de modifier ces règles afin qu'elles ne puissent pas sauter les délimiteurs de mots (les phénomènes intra-mot et inter-mots sont habituellement distincts) et enfin d'ajouter les règles inter-mots (à l'intérieur desquelles les marques de délimitation de mots devraient apparaître explicitement, pour des raisons semblables). Il faut aussi noter que l'aspect de la représentation lexicale que nous avons vu jusqu'ici reflète notre désir de travailler au niveau intuitif plutôt que de passer à celui de la véritable exactitude linguistique.

Il est intéressant de signaler que bien que les règles à deux niveaux soient capables de traiter ces trois phénomènes (réalisation des délimiteurs de mots, élision et contraction), elles semblent plutôt mal convenir à la description du dernier type (contractions). Ceci semble indiquer que, bien que l'orthographe intra-mot *est* régulière (elle peut être décrite adéquatement par des dispositifs à états finis), il existe peut-être des formalismes mieux adaptés pour la décrire que les règles à deux niveaux. Le fait que l'ensemble de règles nécessaires à la description de la contraction de *à le* en *au* pourrait former un transducteur unique et relativement simple semble aller dans le sens de cette idée. L'allure d'un tel autre formalisme est une question qui reste à étudier.

Un autre problème pouvant se produire lors de la conception de règles orthographiques s'appliquant à une langue découle de ce que certaines de ces règles semblent être fondamentalement ambiguës et générer des hypothèses multiples tout aussi bien dans l'analyse que dans la synthèse. Par exemple, en français, la forme de surface *des* peut être analysée soit comme une forme plurielle de l'article indéfini *un*, ou comme la contraction de la préposition *de* et de l'article défini pluriel *les*. D'autre part, cette dernière chaîne lexicale (juxtaposition *de+les* des formes lexicales *de* et *les*) peut produire soit la contraction *des* si *les* est un article, ou simplement *de les* si *les* est un pronom.

Dans un système tel que celui que nous désirons élaborer, les formes de surface produisant des formes lexicales multiples ne sont pas considérées comme problématiques, car des niveaux plus élevés (le lexique, les règles morphologiques, les règles syntaxiques, etc.) devraient pouvoir éliminer les formes lexicales inacceptables. Toutefois, les ambiguités dans 'l'autre sens' présentent des difficultés : par exemple, si l'élément orthographique (c.-à-d. l'élément régissant l'application des règles orthographiques dans la mise en oeuvre du modèle) ne 'sait' pas si une occurrence donnée de la forme lexicale *les* correspond à l'article ou au pronom, il ne peut alors en aucun cas décider si la contraction avec *de* doit s'effectuer ou non. De toute évidence, les règles orthographiques ne peuvent pas fonctionner convenablement si on ne sait pas "quelle règle s'applique à quelle situation". Il semble toutefois que le problème n'a pas uniquement pour origine l'élément orthographique. Si l'on peut tirer un parallèle entre l'orthographe dans un texte écrit et la phonologie dans la langue parlée, l'exemple suivant devrait être suffisant pour illustrer notre point de vue. Dans la phrase française "un marchand de draps anglais", la présence d'une liaison entre *draps* et *anglais* dépend du mot qualifié par l'adjectif *anglais* (soit *marchand* ou *draps*. Oui, bon, cela va un peu loin...). Dans cet exemple, l'application d'une règle phonologique donnée ne dépend pas de l'environnement phonologique immédiat mais de la structure syntaxique

sous-jacente et peut même avoir des 'racines' plus profondes.

Il semble donc raisonnable de supposer que toute la connaissance se rapportant à l'orthographe ne réside pas dans l'élément orthographique lui-même et que nous devons envisager l'orthographe comme étant conditionnée *lexicalement*, *morphologiquement* ou même *syntaxiquement*. Donc, lorsqu'il se produit des ambiguités orthographiques, l'élément orthographique ne 'sait' comment les résoudre que si un élément de niveau plus élevé le lui 'indique'. Il semble que la façon la plus simple de modéliser ce transfert de connaissances dans notre système consiste à annoter directement la chaîne lexicale. Autrement dit, nous pouvons imaginer que la représentation lexicale renferme véritablement beaucoup plus d'informations que l'apparence de surface ne semble l'indiquer. Nous avons déjà mentionné les caractères de délimitation de mot et de délimitation de morphème, mais nous pourrions probablement envisager bien d'autres objets de la chaîne lexicale ayant des réalisations de surface, bien que cela ne soit pas toujours évident. Par exemple, la syntaxe ou le lexique pourraient insérer des *pauses* explicites à certaines positions sélectionnées d'une phrase (ou d'un mot), afin d'inhiber localement l'ensemble ou une partie des règles orthographiques.Une virgule pourrait être une réalisation de surface d'une telle pause 'forte', mais il pourrait également exister des pauses 'plus faibles' ayant des effets orthographiques plus subtils, pouvant par exemple annuler les 'règles d'élision' devant certains mots commençant par un *h* ou encore la 'règle de contraction' de *de les* lorsque *les* est un pronom, etc. On pourrait imaginer d'autres types de marques. Il faut voir ceci comme un moyen de communiquer des informations de plus haut niveau à l'élément orthographique.

Bien entendu, les entrées lexicales, les règles morphologiques et même les règles de grammaire doivent être rédigées en pensant à l'orthographe, et les règles orthographiques doivent tenir compte des informations supplémentaires, mais tout semble possible. Quoi qu'il en soit, la façon dont cela sera réalisé reste à déterminer.

Conclusion

La première partie de cet article était une présentation des modèles de calcul actuels de la morphologie. Un grand nombre des approches sont dérivées du modèle 'à deux niveaux' de Koskenniemi, et, pour cette raison, nous l'avons examiné en premier, puis nous avons continué par une étude rapide de certaines des autres solutions proposées. Dans la seconde partie, nous avons traité la façon dont le modèle de Koskenniemi pourrait être utilisé avec des textes continus pour traiter des phénomènes 'inter-mots' tels que l'élision et la contraction.

Le dernier sujet a été évoqué brièvement et plusieurs questions restent à résoudre, en particulier à savoir si le modèle à deux niveaux est véritablement la meilleure méthode de description de certains de ces phénomènes. La détermination d'un formalisme mieux adapté à cette tâche ainsi que l'étude des systèmes de communication entre l'élément orthographique et des niveaux plus élevés de traitement linguistique semblent tous deux être des sujets intéressants pour des recherches à venir.

Malgré toutes ses faiblesses, nous recommandons l'utilisation du modèle de Koskenniemi comme point de départ pour de tels travaux de recherche, et ceci pour plusieurs rai-

sons pratiques : premièrement, il existe de nombreuses langues naturelles pouvant être décrites de façon relativement adéquate par le formalisme à deux niveaux, et permettant de produire des analyseurs efficaces. Et surtout, le modèle n'est pas orienté vers l'analyse ou la synthèse, et peut être utilisé pour les deux. À notre avis, il s'agit d'un avantage majeur par rapport à des approches telles que celles de Hankamer, Boisen, Cornell et Dolan pour n'en nommer que quelques-unes, qui sont toutes conçues principalement pour effectuer des analyses. Il ne s'agit pas seulement d'une propriété de calcul commode, c'est aussi un critère important d'exactitude linguistique. Enfin, le modèle de Koskenniemi est d'une certaine façon représentatif des approches courantes de la morphologie informatique. Pour nous, cette propriété est importante, car elle donne un certain niveau de généralité à nos résultats : nous estimons que les idées présentées à la section 5 peuvent être appliquées à la plupart des modèles morphologiques à états finis.

Références

[And88] Anderson, Stephen R. (1988), Morphology as a Parsing Problem, dans *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dép. de linguistique, UCLA.

[Bar86] Barton, Jr., G. Edward (1986), *The Computational Complexity of 2-level Morphology*, AI Memo 856, Laboratoire d'IA du MIT.

[Bea86] Bear, John (1986), A Morphological Recognizer with Syntactic and Phonological Rules, *COLING 86* (*Proceedings of the 11th International Conference on Computational Linguistics*), pp. 272-276.

[Bea88] Bear, John (1988), Morphology with Two-level Rules and Negative Rule Features, *COLING 88*, pp. 28-31.

[Boi88a] Boisen, Sean (1988), Parsing Morphology using Definite Clauses, dans *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dép. de linguistique, UCLA.

[Boi88b] Boisen, Sean (1988), Pro-KIMMO: a Prolog Implementation of Two-level Morphology,dans *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dép. de linguistique, UCLA.

[Cor88] Cornell, Thomas L. (1988), IceParse: A Model of Inflectional Parsing and Word Recognition for Icelandic Ablauting Verbs, dans *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dép. de linguistique, UCLA.

[DKK87] Dalrymple, Mary, Ronald M. Kaplan, Lauri Karttunen, Kimmo Koskenniemi, Sami Shaio et Michael Westcoast (1987), *Tools for Morphological Analysis*, rapport n CSLI-87-108, CSLI, Université Stanford.

[Dol88] Dolan, William B. (1988), A Syllable-based Parallel Processing Model for Parsing Indonesian Morphology, dans *Morphology as a Computational Problem*, UCLA Occasional Papers #7: Working Papers in Morphology, Dép. de linguistique, UCLA.

[Han86] Hankamer, Jorge (1986), Finite-state Morphology and Left-to-right Phonology, *West Coast Conference on Formal Linguistics*, 5, pp. 41-52.

[HU79] Hopcroft, John E. et Jeffrey D. Ullman (1979), *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley.

[Kar83] Karttunen, Lauri (1983) KIMMO: A General Morphological Processor, *Texas Linguistic Forum* #22, pp. 165-186.

[Kos83] Koskenniemi, Kimmo (1983), *Two-level Morphology: a General Computational Modem for Word-form Recognition and Production*, Publication 11, Dép. de linguistique générale, Université d'Helsinki.

[KW83] Karttunen, Lauri et Kent Wittenberg (1983), Two-level Morphological Analysis of English, *Texas Linguistic Forum* #22, pp. 217-228.